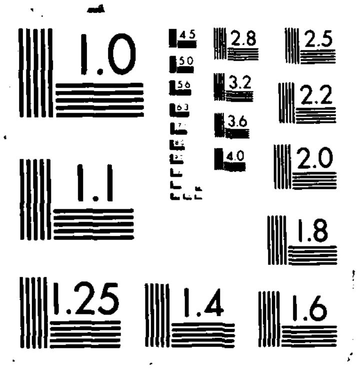
AD-A181 111   A LARGE-SIGNAL ANALYSIS PROGRAM FOR HELIX   1/1
TRAVELING-WAVE TUBSS (U) UTAH UNIV SALT LAKE CITY DEPT
OF ELECTRICAL ENGINEERING T A BRUNASSO FEB 87
UNCLASSIFIED   UTEC-MD-86-063 RADC-TR-87-25   F/G 9/1   NL


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

12

RADC-TR-87-25
Final Technical Report
February 1987

A LARGE-SIGNAL ANALYSIS PROGRAM FOR HELIX TRAVELING-WAVE TUBES

University of Utah

Theresa A. Brunasso

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-87-25 has been reviewed and is approved for publication.

APPROVED: Andrew E Chrostowski

ANDREW E. CHROSTOWSKI, 1Lt, USAF
Project Engineer

APPROVED: Frank J Rehm

FRANK J. REHM
Technical Director
Directorate of Surveillance

FOR THE COMMANDER: John A. Ritz

JOHN A. RITZ
Directorate of Plans & Programs

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (OCTP) Griffiss AFB NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

AD A181 111

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS N/A
2a. SECURITY CLASSIFICATION AUTHORITY N/A 2b. DECLASSIFICATION / DOWNGRADING SCHEDULE N/A	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution unlimited
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UTEC MD-86-063	5. MONITORING ORGANIZATION REPORT NUMBER(S) RADC-TR-87-25
6a. NAME OF PERFORMING ORGANIZATION University of Utah 6b. OFFICE SYMBOL *(If applicable)*	7a. NAME OF MONITORING ORGANIZATION Rome Air Development Center (OCTP)
6c. ADDRESS *(City, State, and ZIP Code)* Department of Electrical Engineering Salt Lake City UT 84112	7b. ADDRESS *(City, State, and ZIP Code)* Griffiss AFB NY 13441-5700
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Rome Air Development Center 8b. OFFICE SYMBOL *(If applicable)* OCTP	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F30602-82-C-0161
8c. ADDRESS *(City, State, and ZIP Code)* Griffiss AFB NY 13441-5700	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
61102F	2305	J9	16

11. TITLE *(Include Security Classification)*
A LARGE-SIGNAL ANALYSIS PROGRAM FOR HELIX TRAVELING-WAVE TUBES

12. PERSONAL AUTHOR(S)
Theresa A. Brunasso

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(Year, Month, Day)*	15. PAGE COUNT
Final	FROM Sep 82 TO Oct 86	February 1987	96

16. SUPPLEMENTARY NOTATION Research accomplished in conjunction with Air Force Thermionics Engineering Research Program (AFTER) ,AFTER-17. Theresa A. Brunasso was an AFTER student from Teledyne MEC. This report was submitted in partial fulfillment of the (over)

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
09	03	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Helix Traveling-Wave Tubes
One-dimensional Modeling
Large Signal Analysis
Computer Program
TWT Modeling Code

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

A one-dimensional large-signal analysis computer code for Helix traveling-wave tubes was developed. The program is based on the theory developed by J.E. Rowe and M.K. Scherba. The program was written in FORTRAN 77 and implemented on an HP-1000 computer at Teledyne MEC in Palo Alto CA. The program is well documented and can accommodate attenuators, severs, velocity changes, linear velocity tapers, and multiple input signals.

Two Teledyne MEC tubes were simulated with the program and the results were in very good agreement with experimental data.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Andrew E. Chrostowski, 1Lt, USAF	22b. TELEPHONE *(Include Area Code)* (315) 330-4381 22c. OFFICE SYMBOL RADC (OCTP)

Previous editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

Block 16. Supplementary Notation (Cont'd)
requirements of the degree of Electrical Engineer.

ACKNOWLEDGMENTS

This work was pursued under the joint sponsorship of the United States Air Force and Teledyne MEC in conjunction with the University of Utah under the Air Force Thermionic Engineering and Research (AFTER) Program. The author wishes to thank Professors Richard W. Grow and J. Mark Baird at the University of Utah for their suggestions and guidance. Thanks are also due to several people in the Advanced Development Group at Teledyne MEC. Phil Lally, Larry Wood, and Eric Nettesheim provided helpful discussions about large-signal theory. Keith Winnovich and Lou Lau supplied the experimental data used to evaluate the program. The work described in this report is heavily dependent upon the work of Rick Baer while he was at Teledyne MEC. Finally, a special thanks goes to Dr. Brian Boe for his assistance and encouragement throughout the course of the project.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS AND TABLES

Table Page

I. INTRODUCTION

Large-signal analysis programs are widely used by the microwave tube community to predict the performance of helix traveling-wave tubes. These nonlinear programs allow an engineer to optimize a tube design without building several prototype tubes. Unfortunately, currently available programs are complex and poorly documented. The purpose of this project is to provide a structured, well documented large-signal analysis program to the scientific and industrial community.

The program, TWT1D, is based on the one-dimension nonlinear theory developed by M. K. Scherba and J. E. Rowe.[1] The program was written in FORTRAN77 and implemented on an HP-1000 computer at Teledyne MEC in Palo Alto, California. Complete documentation for the program, including a source code listing and a User's Manual, is given in this report.

II. LARGE-SIGNAL THEORY

The large-signal theory of traveling-wave amplifiers differs from small-signal theory in several ways. Perhaps the most significant difference is the way each theory models the electron beam. In small-signal theory, the beam is modeled as drifting charged fluid. Its charge density and electron velocity are single valued functions of distance, and electrons cannot overtake one another. This treatment of the electron beam is called an Eulerian formulation. By comparison, the Lagrangian analysis used in large-signal theory breaks the beam up into representative charge groups. This approach allows charge density and electron velocity to be multivalued functions, and electrons to overtake one another. Another difference between the theories is the way they treat the force and continuity equations. The equations are linearized in small-signal theory, so the theory is limited to circuits with small "c" and small input signals. Linearizing the equations allows them to be solved analytically. The solution is a single number which gives the gain/unit length for the tube. By contrast, the large-signal theories have no analytic solution and must be solved on a computer. The resulting solution is a profile of power versus axial position.

The large-signal theory offers many advantages over small-signal theory. Since the equations are nonlinear, harmonics can be generated and followed down the tube. This allows the effects of intermodulation to be studied. Unlike small-signal theory, large-signal theory can be used to predict the saturation power and efficiency of a tube. In addition, using a Lagrangian analysis means that electron trajectories can be followed down the tube so beam bunching can be observed. Finally, the theory can accommodate attenuators, severs, and velocity changes.

The first attempt at developing a large-signal theory for helix traveling-wave tubes was made by A. T. Nordsieck[2] in 1953. He was the first to use Lagrangian analysis to model the interaction between the circuit and the electron beam for a single input signal. However, his theory was subject to several limitations. To begin with, he assumed that space-charge forces could be neglected. Secondly, he limited his theory to lossless circuits with small C. A year later, H. C. Poulter[3] extended the theory to include space-charge forces and to account for circuit loss and finite C. Two years after that, J. E. Rowe[4] amended the theory to allow for circuits with a large C. Finally, in 1971, M. K. Scherba and J. E. Rowe[1] expanded the theory to include multiple input signals. It is this theory that is the basis for the program TWT1D.

In Rowe's model, the helix is represented by a lumped element transmission line, and the electron beam is represented by disks of charge. These charge groups move along the circuit inducing charges in the circuit. The RF field in the circuit interacts with the electron disks and causes them to bunch. This model was first used by Brillouin,[5] and is schematically represented in Fig. 1.

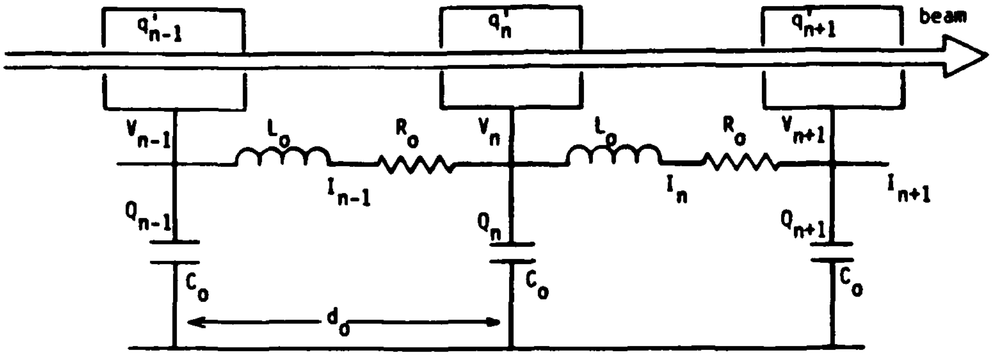


Fig. 1. Schematic model of a helix with an electron beam.

where

d_0 = length of section

$R_0 = Rd_0$ = resistance/section

$L_0 = Ld_0$ = inductance/section

$C_0 = Cd_0$ = capacitance/section

I_n = current in L_0 of section n

V_n = potential on capacitor n

$Q_n = C_0V_n$ = charge on capacitor n

$q'_n = \rho_n d_0$ = charge in the beam in section n

The circuit equation for this transmission line is

$$\frac{\partial^2 V}{\partial t^2} + \frac{R}{L}\frac{\partial V}{\partial t} - \nu_0^2 \frac{\partial^2 V}{\partial z^2} = \nu_0 Z_0 \left(\frac{\partial^2 \rho}{\partial t^2} + \frac{R}{L}\frac{\partial \rho}{\partial t}\right) \qquad (1)$$

where the following definitions have been used:

$v_0 \triangleq \sqrt{1/LC}$ = the characteristic phase velocity of the circuit

$Z_0 \triangleq \sqrt{L/C}$ = the characteristic impedance of the circuit

To express this equation in Pierce[6] notation, the following relation is used:

$$R/L = 2\omega Cd \tag{2}$$

where C is Pierce's gain parameter and d is his loss parameter. The equation then becomes

$$\frac{\partial^2 V}{\partial t^2} - v_0^2 \frac{\partial^2 V}{\partial z^2} + 2\omega Cd \frac{\partial V}{\partial t} = v_0 Z_0 \left(\frac{\partial^2 \rho}{\partial t^2} + 2\omega Cd \frac{\partial \rho}{\partial t} \right) \tag{3}$$

For multiple signals, there will be a circuit equation for each signal:

$$\frac{\partial^2 V_n}{\partial t^2} - v_{0n}^2 \frac{\partial^2 V}{\partial z^2} + 2\omega_n C_n d_n \frac{\partial V_n}{\partial t} = v_{0n} Z_{0n} \left(\frac{\partial^2 \rho_n}{\partial t^2} + 2\omega_n C_n d_n \frac{\partial \rho_n}{\partial t} \right) \tag{4}$$

A. Phase Relation Equation

In his Lagrangian analysis, Nordsieck introduced two independent variables. They are the normalized distance, y, and the entry phase of the fundamental component of the input signal, ϕ_{01}. They are defined as

$$y \triangleq \frac{C_1 \omega_1}{u_0} z \tag{5}$$

$$\phi_{0_1} \triangleq \frac{\omega_1 z}{u_0} = - \omega_1 t_0 \tag{6}$$

where ω_1 is the fundamental frequency in the radians, and C_1 is the gain parameter for the fundamental frequency. u_0 is the dc stream velocity in m/s, and z is the axial position on the tube measured in meters. Using these new variables, the circuit equation becomes

$$\frac{\partial^2 V_n(y,t)}{\partial t^2} - C_1^2\omega_1^2\left(\frac{v_{0n}}{u_0}\right)^2 \frac{\partial^2 V_n(y,t)}{\partial y^2} + 2\omega_n C_n d_n \frac{\partial V_n(y,t)}{\partial t}$$

$$= v_{0n}Z_{0n}\left(\frac{\partial^2 \rho_n(y,t)}{\partial t^2} + 2\omega_n C_n d_n \frac{\partial \rho_n(y,t)}{\partial t}\right) \tag{7}$$

Nordsieck also introduced the following dependent variables:

$$\Phi_n(y,\Phi_{01}) = \omega_n\left(\frac{z}{u_0} - t\right) - \theta_n(y) = \frac{\omega_n}{\omega_1 C_1} y - \omega_n t - \theta_n(y) \tag{8}$$

$$u_z = \left.\frac{dz}{dt}\right|_{t,z_o} = \frac{u_0}{C_1\omega_1}\frac{dy}{dt} = u_0\left[1 + 2C_1 u(y,\Phi_{01})\right] \tag{9}$$

where Φ_n is the instantaneous charge group phase in radians, and θ_n is the phase lag in radians of the circuit wave relative to a traveling wave with phase velocity u_0. This phase lag is due to beam loading as the wave gets energy from the beam. u_z is the electron velocity normalized to the initial average electron velocity, u_0. Taking the derivative with respect to time of Eq. 8 and substituting Eq. 9 yields

$$\frac{\partial\theta_n(y)}{\partial y} + \frac{\partial\Phi_n(y,\Phi_{01})}{\partial y} = \frac{\omega_n}{\omega_1 C_1}\left[1 - \frac{1}{1 + 2C_1 u(y,\Phi_{01})}\right] \tag{10}$$

This is the first of Rowe's large-signal equations. It is the velocity-phase equation relating u_z, θ_n, and Φ_n.

B. Circuit Wave Equations

The next two large-signal equations arise from the circuit equation. To evaluate the circuit equation, define the RF voltage,

$$V_n\left(y,\Phi_{01}\right) = \sum_n V(y)\; V\left(\Phi_n\right) \tag{11}$$

where

$$V(y) = \frac{Z_{01} I_0}{C_1} A_n(y) \quad \text{and} \quad V\left(\Phi_n\right) = e^{-j\Phi_n} \tag{12}$$

Z_{01} is the interaction impedance for the fundamental frequency, and $A_n(y)$ is the normalized circuit voltage amplitude for the nth frequency. Substituting the derivataives of Eq. 11 into Eq. 4 yields an equation in terms of sin Φ_n and cos Φ_n. Since these terms are mutually orthogonal, they can be separated and the original circuit equation can be expressed as two equations,

$$\frac{d^2 A_n(y)}{\partial y^2} - A_n(y)\left[\left(\frac{\omega_n}{\omega_1 C_1} - \frac{d\theta_n}{dy}\right)^2 - \left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_n b_n}{C_1}\right)^2\right] = -\frac{R_n \cos\,(\rho)}{\omega_1^2 C_1 Z_{01} I_0}\left(\frac{u_0}{\nu_{0n}}\right)^2 \tag{13}$$

$$A_n(y)\left[\frac{d^2\theta_n}{dy^2} - 2C_n d_n\left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_n b_n}{C_1}\right)^2\right] - 2\,\frac{dA_n(y)}{dy}\left(\frac{\omega_n}{\omega_1 C_1} - \frac{d\theta_n}{dy}\right) = -\frac{R_n \sin\,(\rho)}{\omega_1^2 C_1 Z_{01} I_0}\left(\frac{u_0}{\nu_{0n}}\right)^2 \tag{14}$$

where R_n sin (ρ) and R_n cos (ρ) are the sine and cosine parts of the space-charge expression, and b_n is Pierce's velocity parameter. Now the problem remains of expressing the space-charge expression in terms of sin Φ_n and cos Φ_n. Since the beam-charge density is rich in harmonics, it is convenient to expand it into a Fourier series:

$$\rho_n(y,\Phi_n) = \sum_{m=1}^{\infty}\left[\frac{\sin m\Phi_n}{\pi}\int_0^{2\pi}\rho \sin m\Phi_n d\Phi_n\right] + \sum_{m=1}^{\infty}\left[\frac{\cos m\Phi_n}{\pi}\int_0^{2\pi}\rho \cos m\Phi_n d\Phi_n\right] \qquad (15)$$

The ρ inside the integral may be determined from continuity arguments:

$$\rho = \frac{I_0}{u_0}\left|\frac{\partial\Phi_{01}}{\partial\Phi_n}\right|\frac{1}{1 + 2C_1 u(y,\Phi_{01})} \qquad (16)$$

Using this expression inside the Fourier integral yields

$$\rho_n(y,\Phi_{0n}) = \frac{I_0}{u_0\pi}\sum_m\left[\sin m\Phi_n \int_0^{2\pi}\frac{\sin m\Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1 u(y,\Phi'_{01})}\right.$$

$$\left. + \cos m\Phi_n \int_0^{2\pi}\frac{\cos m\Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1 u(y,\Phi'_{01})}\right] \qquad (17)$$

The summation over m represents the sum of all harmonic frequencies for each of the n fundamental frequencies. Assuming that the circuit beam interactions at the harmonic frequencies are negligible and keeping only the m = 1 terms,

$$\rho_n(y,\Phi_{0n}) = \frac{I_0}{u_0\pi}\left[\sin\Phi_n \int_0^{2\pi} \frac{\sin\Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1u(y,\Phi'_{01})} + \cos\Phi_n \int_0^{2\pi} \frac{\cos\Phi_n(y,\Phi'_{01})d\Phi'_{01})}{1 + 2C_1u(y,\Phi'_{01})}\right] \tag{18}$$

These values may now be substituted into the circuit equation to get Rowe's next two large-signal equations:

$$\frac{d^2A_n(y)}{dy^2} - A_n(y)\left[\left(\frac{\omega_n}{\omega_1C_1} - \frac{d\theta_n}{dy}\right)^2 - \left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_nb_n}{C_1}\right)^2\right]$$

$$= \frac{Z_{0n}}{Z_{01}}\left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_nb_n}{C_1\tau}\right)\left[2C_nd_n \int_0^{2\pi} \frac{\sin\Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1u(y,\Phi'_{01})} + \int_0^{2\pi} \frac{\cos\Phi_n(y,\Phi'_{01})d\Phi'_{01})}{1 + 2C_1u(y,\Phi'_{01})}\right] \tag{19}$$

$$A_n(y)\left[\frac{d^2\theta_n}{dy^2} - 2C_nd_n\left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_nb_n}{C_1}\right)^2\right] - 2\frac{dA_n(y)}{dy}\left(\frac{\omega_n}{\omega_1C_1} - \frac{d\theta_n}{dy}\right)$$

$$= \frac{Z_{0n}}{Z_{01}}\left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_nb_n}{C_1\pi}\right)\left[\int_0^{2\pi} \frac{\sin\Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1u(y,\Phi'_{01})} - 2C_nd_n \int_0^{2\pi} \frac{\cos\Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1u(y,\Phi'_{01})}\right] \tag{20}$$

Equation 19 is the circuit wave amplitude equation, and Eq. 20 is the circuit wave phase equation.

C. Electron Force Equation

Rowe's final large-signal equation is derived from the Lorentz force equation,

$$\frac{d^2 z}{dt^2} = |\eta| \left(\frac{\partial V_c}{\partial z} + \frac{\partial V_{sc}}{\partial z} \right) \qquad (21)$$

where V_{sc} and V_c are the space-charge and circuit terms, respectively. Using Eq. 9 and the chain rule for derivatives and substituting into Eq. 21 yields

$$\frac{\partial u(y,\Phi_{01})}{\partial y} \left[1 + 2C_1 u(y,\Phi_{01})\right] = \frac{|\eta|}{2C_1 u_0^2} \left\{ \frac{Z_{01} I_0}{C_1} \sum_n \left[\frac{dA_n(y)}{dy} \cos \Phi_n - A_n(y) \sin \Phi_n \left(\frac{\omega_n}{\omega_1 C_1} - \frac{d\theta_n}{dy} \right) \right] - \frac{1}{\omega_1} E_{sc} \right\} \qquad (22)$$

where E_{sc} is the space-charge field. The constant in front of the circuit wave term can be simplified using the definition of Pierce's gain parameter,

$$\frac{\partial u(y,\Phi_{01})}{\partial y} \left[1 + 2C_1 u(y,\Phi_{01})\right] = C_1 \sum_n \left[\frac{dA_n(y)}{dy} \cos \Phi_n - A_n(y) \sin \Phi_n \left(\frac{\omega_n}{\omega_1 C_1} - \frac{d\theta_n}{dy} \right) \right] - \frac{|\eta|}{2C_1^2 \omega_1 u_0} E_{sc} \qquad (23)$$

Rowe[4] derived a space-charge expression,

$$E_{sc} = - \frac{2\omega_1 u_0}{|\eta| (1 + C_n b_n)} \left(\frac{\omega_p}{\omega_1} \right)^2 \int_0^{2\pi} \frac{F(\Phi - \Phi'_{01}) d\Phi'_{01}}{1 + 2C_1 u(y, \Phi_{01})} \qquad (24)$$

where $F(\phi - \phi'_{01})$ is a space-charge weighting function derived by Poulter.[3] Rowe developed a closed form expression for this function:

$$F(\phi - \phi'_{01}) = \frac{1}{2\pi}\left[\frac{\pi - (\phi - \phi'_{01})}{2} - 2\tan^{-1}\frac{\sin(\phi - \phi'_{01})}{e^{\beta b' f(a'/b')} - \cos(\phi - \phi'_{01})} + \tan^{-1}\frac{\sin(\phi - \phi'_{01})}{e^{2\beta b' f(a'/b')} - \cos(\phi - \phi'_{01})}\right]$$

$$\text{for } 0 < (\phi - \phi'_{01}) < 2\pi \qquad (25)$$

where a' and b' are the helix and beam radii, respectively, and

$$f(a'/b') = \frac{\partial \ln(1 - R_n)}{\partial \beta b'} \qquad \text{at constant } a'/b' \qquad (26)$$

R_n is the electron plasma frequency reduction factor,

$$R_n^2 \triangleq 1 - \frac{\beta b'}{I_0(\beta a')}\left[I_1(\beta b')\,K_0(\beta a') + I_0(\beta a')\,K_1(\beta b')\right] \qquad (27)$$

where I_n and K_n are modified Bessel functions. Substituting the space-charge expression into Eq. 22 yields the fourth and final large-signal equation.

$$\frac{\partial u(y,\phi_{01})}{\partial y}\left[1 + 2C_1 u(y,\phi_{01})\right] = C_1 \sum_n \left[\frac{dA_n(y)}{dy}\cos\phi_n - A_n(y)\sin\phi_n\left(\frac{\omega_n}{\omega_1 C_1} - \frac{d\theta_n}{dy}\right)\right] + \left(\frac{1}{1 + C_n b_n}\right)\left(\frac{\omega_\rho}{\omega_1 C_1}\right)^2 \int_0^{2\pi} \frac{F(\phi - \phi'_{01})\,d\phi'_{01}}{1 + 2C_1 u(y,\phi_{01})}$$

(28)

D. Initial Conditions

The solution of Rowe's large-signal equations may be treated as an initial-value problem if it is assumed that the helix is terminated in its characteristic impedance so that there are no reflections at the output. The initial conditions for the beam and circuit wave are derived in this section. In deriving these conditions, it was assumed that the electron beam is unmodulated at the input.

At the input to the tube (y = 0), there is no RF interaction, so the second derivatives are zero. Also, since the beam is unmodulated at the input, the space-charge terms are zero. Thus, the circuit wave phase equation becomes

$$\frac{d\theta_n}{dy} = - \frac{\omega_n C_n b_n}{\omega_1 C_1} \qquad (29)$$

Substituting this into the circuit wave amplitude equation at the input yields

$$\frac{dA_n(y)}{dy} = - A_n(y) C_n d_n \left(\frac{\omega_n}{\omega_1 C_1} \right) (1 + C_n b_n) \qquad (30)$$

The unmodulated beam at the input is represented by uniformly distributing the electron charge groups over one cycle of the fundamental frequency. For i charge groups, the initial phases are given by

$$\Phi_{0,j} = \frac{2\pi j}{i} \qquad j = 0, 1, 2, \ldots, i \qquad (31)$$

The velocity of the beam at the input is equal to the dc stream velocity, so that

$$1 + 2C_1 u(0,\phi_{0,j}) \equiv 1 \qquad (32)$$

III. ATTENUATORS AND SEVERS

Virtually all helix traveling-wave tubes use attenuators and/or severs to prevent oscillations. Therefore, any program used for the design of these tubes must be able to model circuit loss.

The simplest way to model loss in a traveling-wave tube is to increase Pierce's loss parameter, d. This is the method recommended by Rowe.[7] Figure 2 shows the result of various Gaussian shaped attenuators on output power, and Fig. 3 shows the result for a region of uniform loss. As we would expect, the attenuators increase the saturation length without decreasing the saturation power. (This would not be the case if the attenuator was too close to the output.)

Unfortunately, there is a problem with this method of modeling loss in a circuit. It will lead to instabilities for large values of loss. This is due to the fact that both the propagation constant, β, and the impedance, Z, vary with the resistance in the circuit. The change in β causes the circuit wave to lose synchronism with the beam, and the change in Z causes reflections in the attenuator due to the impedance mismatch. The result of these problems is that the circuit wave phase angle oscillates in a region of heavy attenuation. This is illustrated in Fig. 4. The higher the loss and/or the longer the region of attenuation, the greater the oscillations. In order to work around this problem in the model, attenuators are replaced by severs when their loss exceeds a certain level.

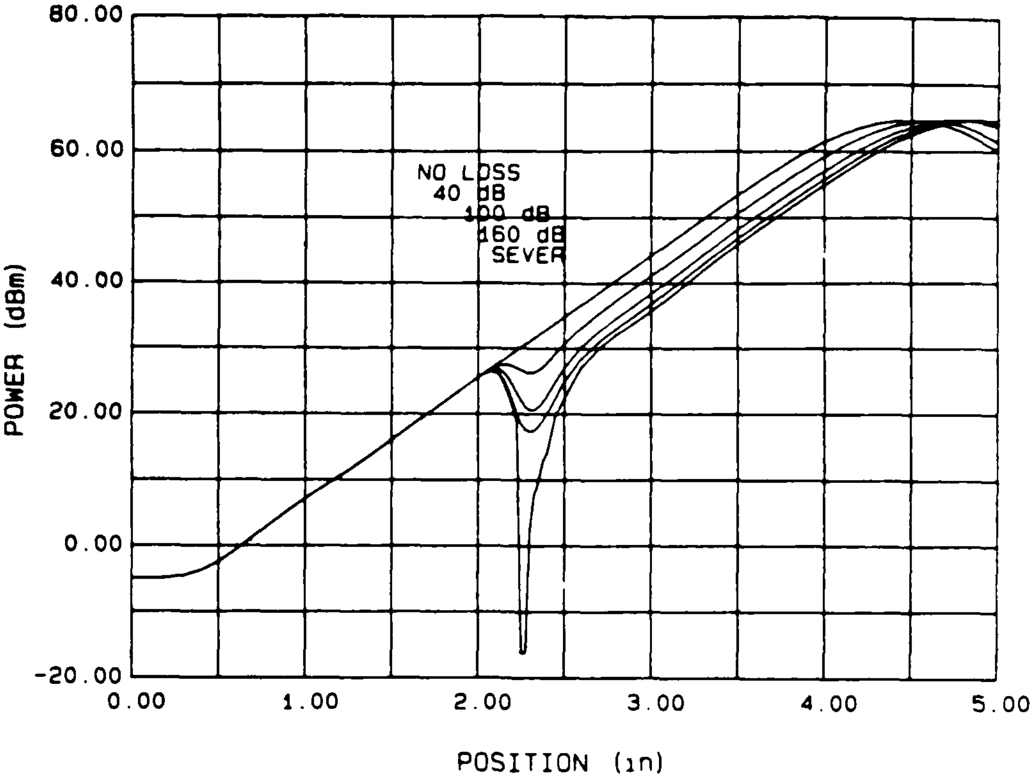


Fig. 2. Gaussian shaped attenuators.

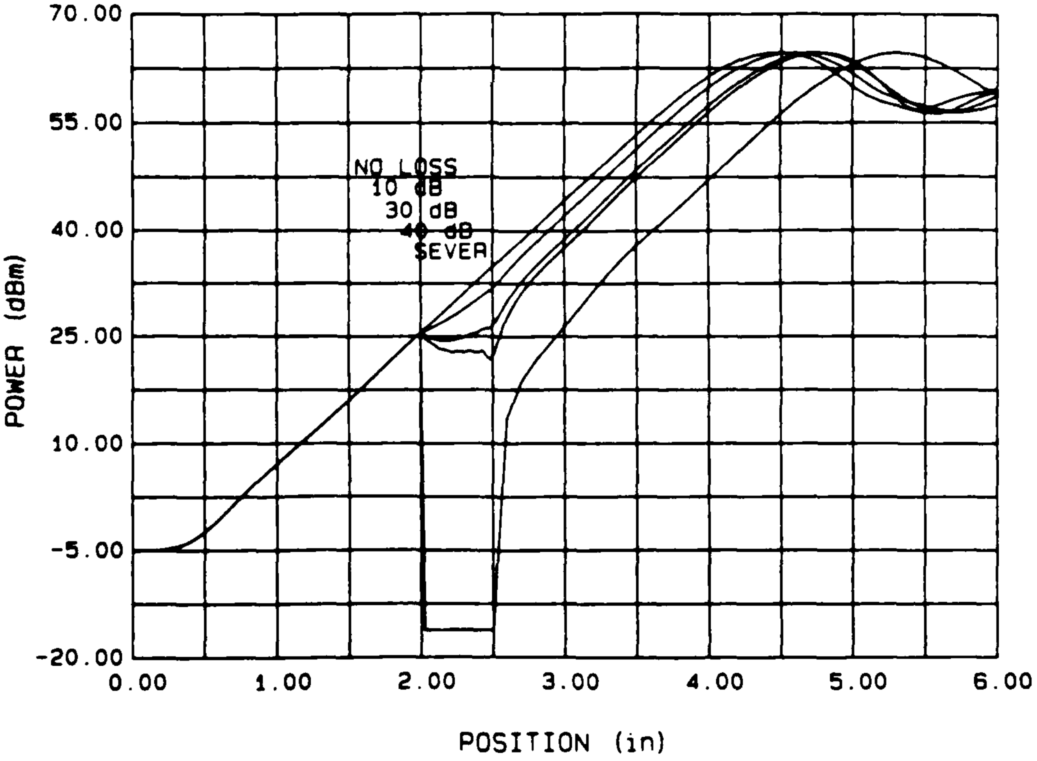


Fig. 3. Uniform attenuators.

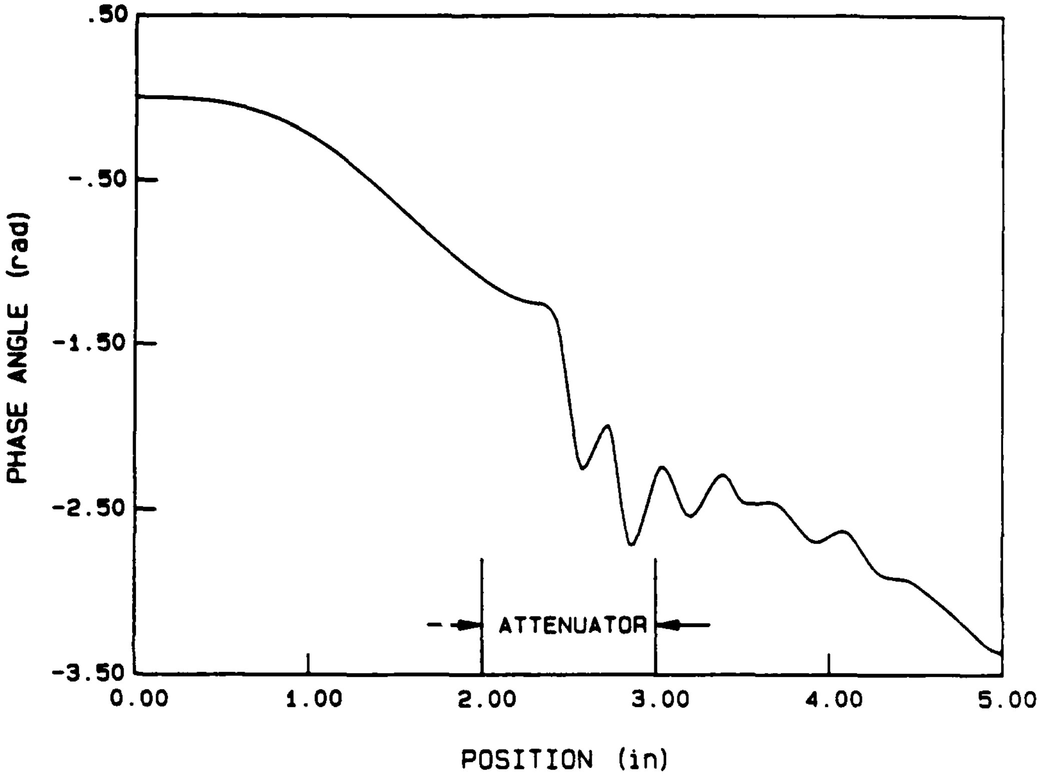


Fig. 4. Oscillation of circuit-wave phase in an attenuator.

In fact, the effect of a strong attenuator is much the same as that of a sever. In a sever, the circuit wave is eliminated and the beam drifts down the tube affected only by the space-charge forces. Similarly, a strong attenuator will diminish the circuit wave so much that the electron beam is unaffected by it. Thus, it is reasonable to model a region of strong attenuation with a sever.

The equations inside the sever are simply Rowe's large-signal equations with the circuit wave terms removed,

$$\frac{\partial u(y,\Phi_{01})}{\partial y}\left[1 + 2C_1 u(y,\Phi_{01})\right] = \left(\frac{\omega_p}{\omega_1 C_1}\right)^2 \left(\frac{1}{1 + C_n b_n}\right) \int_0^{2\pi} \frac{F(\Phi - \Phi'_{01})d\Phi'_{01}}{1 + 2C_1 u(y,\Phi_{01})} \tag{33}$$

$$\frac{\partial \Phi_n(y,\Phi_{01})}{\partial y} = 2\left(\frac{\omega_n}{\omega_1}\right)\left(\frac{u(y,\Phi_{01})}{1 + 2C_1 u(y,\Phi_{01})}\right) \tag{34}$$

The first equation is the electron force equation, and the second is the electron phase equation. Once inside a sever, the circuit wave phase, the circuit wave amplitude, and their derivatives are zero. The remaining two drift tube equations are integrated through the sever.

At the end of the sever, the circuit wave must be restarted. Rowe[7] suggests that the following initial conditions be used: $A_n(y) = 0$, $\theta_n(y) = 0$, and $d\theta_n(y)/dy = -\omega_n b_n c_n/\omega_1 C_1$. However, the last condition requires the beam to be unmodulated (Eq. 28). In fact, the beam is highly modulated in the sever, and it is this modulation that restarts the circuit wave. I suggest the following conditions for the end of a sever. Since there is no RF interaction in the sever, the second derivatives are zero. So the circuit wave phase equation becomes

$$\frac{d\theta_n(y)}{dy} = \frac{\omega_n}{\omega_1 C_1} - \left(\frac{\omega_n}{\omega_1}\right)\left[\left(\frac{1 + C_n b_n}{C_1}\right)^2 - \frac{1}{A_n(y)}\left\{\left(\frac{Z_{0n}}{Z_{01}}\right)\left(\frac{\omega_n}{\omega_1}\right)\left(\frac{1 + C_n b_n}{C_1 \pi}\right)\right.\right.$$
$$\left.\left.\left[\int_0^{2\pi} \frac{\cos \Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1 u(y,\Phi'_{01})} + 2C_n d_n \int_0^{2\pi} \frac{\sin \Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1 u(y,\Phi'_{01})}\right]\right\}\right] \tag{35}$$

and the circuit wave amplitude equation becomes

$$\frac{dA_n(y)}{dy} = \left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_n b_n}{C_1}\right)\left(\frac{d\theta_n}{dy} - \frac{\omega_n}{\omega_1 C_1}\right)^{-1}\left\{A_n(y)C_n d_n\left(\frac{1 + C_n b_n}{C_1}\right)\right.$$
$$\left. + \left(\frac{Z_{0n}}{Z_{01}}\right)\left(\frac{\omega_n}{\omega_1}\right)\left(\frac{1}{2\pi}\right)\left[\int_0^{2\pi} \frac{\sin \Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1 u(y,\Phi'_{01})} - 2C_n d_n \int_0^{2\pi} \frac{\cos \Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1 u(y,\Phi'_{01})}\right]\right\} \tag{36}$$

Note that Eq. 35 has the circuit wave amplitude, $A_n(y)$, in the denominator. This prevents us from assigning a value of zero to this variable at the end of a sever. Instead of zero, the maximum of 1 percent of the input value or 1 percent of the present value of A_n is used. This value is large enough to prevent numerical overflow, yet small enough to realistically model a sever. The power and circuit wave phase are plotted using both Rowe's and the new initial conditions at the end of a sever in Figs. 5 and 6. As the figures show, the new initial conditions greatly reduce the oscillations in the circuit phase plot and eliminate them completely in the power plot.

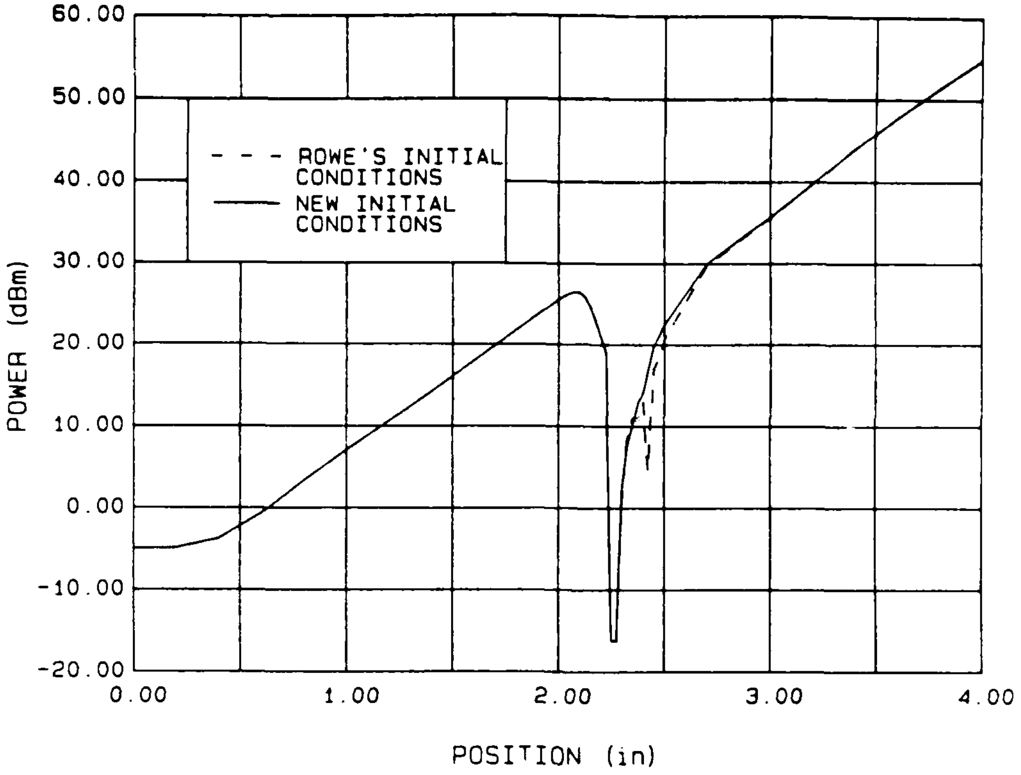


Fig. 5. Power plot of a Gaussian shaped sever.

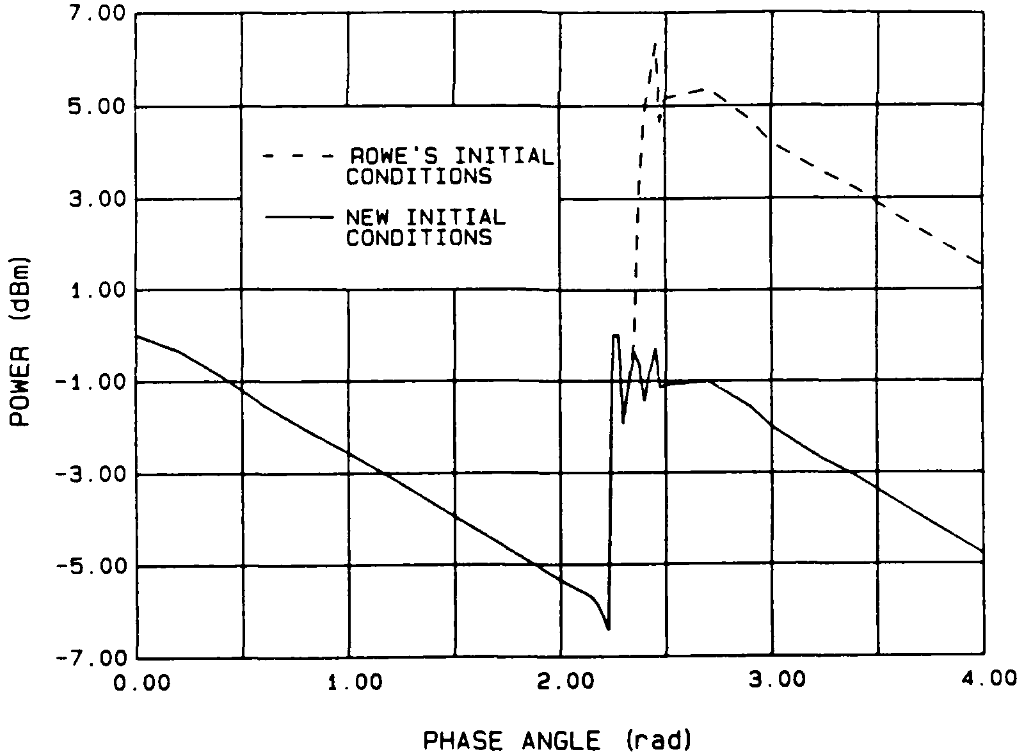


Fig. 6. Phase plot of a Gaussian shaped sever.

Figure 7 shows the result of an abrupt sever. Note that after the sever, the power rapidly grows to a level approximately 6 dB below what it would have been without the sever. This result is in agreement with Pierce[5] and experimental results.

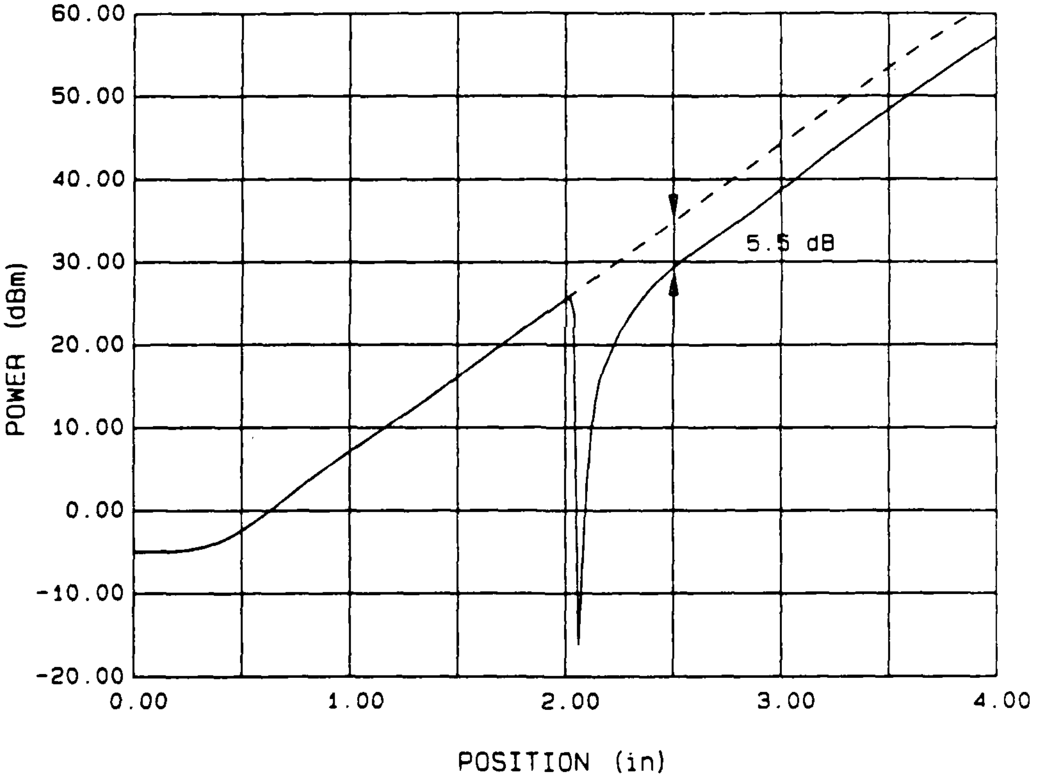


Fig. 7. Power plot of an abrupt sever.

IV. COMPUTER IMPLEMENTATION OF ROWE'S EQUATIONS

This section describes how the large-signal equations derived in the previous sections are set up and solved in TWT1D. In addition, it shows how the program is organized, and how attenuators, severs, velocity steps, and tapers are handled. TWT1D is composed of a main program block and 23 subprograms. The program blocks can be grouped according to the functions they serve, as illustrated in Table 1.

Table 1. TWT1D program blocks organized by function.

Function	Program Block
PROGRAM CONTROL	TWT1D ZSORT EVENT INTSUB
ROWE'S LARGE SIGNAL EQUATIONS	DIFFEQ DIFF DRFTEQ DRIFT
INTEGRALS IN ROWE'S EQUATIONS	FINT INT1 INT2
NUMERICAL ANALYSIS	DESOLV STEP INTRP MACHIN BESSEL
OUTPUT	OUTPUT OUTRS
CIRCUIT CHANGES	ATTEN CHANGE
DIAGNOSTICS	ENERGY BADIN
ROWE'S SPACE-CHARGE EXPRESSION	SPACE SIMINT

A. Program Control

TWT1D is set up to integrate Rowe's large-signal equations down a tube, with stops for attenuators, velocity changes, and restart data printouts. At each of these stops, the appropriate subroutine is called to implement the change in parameters or the printout. The flow of the program is controlled by the main program block and subroutines ZSORT, EVENT, and INTSUB.

The main program block is divided into 9 sections. The first section sets up the arrays, variables, and constants used in the main program and various subprograms. The second section controls the terminal display and keyboard input when the program is run. The third section reads the data from the input data file. In addition, the attenuators and velocity tapers are set up in this section. The next section converts the input data into Pierce parameters and normalized Rowe variables. Section 5 reads the restart data or, if no restart data are available, initializes the beam and circuit wave. In Section 6, the output data file is opened, and the Pierce parameters computed in Section 4 are printed out. In order to facilitate the program flow, Section 7 puts all events that require stopping the integration into a single list, sorted according to their position along the circuit. These events are restart data printouts, attenuators, and velocity steps and tapers. The eighth section controls the actual flow of the program. It moves down the axial position list, integrating Rowe's large-signal equations. At each event, the integration is stopped, the type of event is determined, and the appropriate subroutine called. The last section computes and prints out the saturation power, gain, efficiency, and length. The output files are then closed, and program execution stops.

Subroutine ZSORT does the actual sorting of the axial position list. It creates two lists: one of stop positions, ZSTOP(i), and one of event type codes corresponding to the stop positions, SWLIST(i). Each event type is assigned a specific prime number code. When more than one event occurs at a position, the event type codes are multiplied. Thus, the numbers can be factored to determine which events occur at each stop position.

Subroutine EVENT checks the event code at each stop position to determine which events occur. The event codes used are: 2 Print restart data; 3 Start attenuator; 5 Stop attenuator; 7 velocity change.

The control code in the main program block stops the integration of the large-signal equations for attenuators, velocity changes, and restart data printouts. However, the integration must also be stopped for the output printouts. Subroutine INTSUB accomplishes this by breaking up tube sections between successive stop events into print intervals. At each print interval, subroutines are called to integrate Rowe's equations and print the output data. In addition, INTSUB ensures that the equations are integrated to the end of the section.

B. Rowe's Large-Signal Equations

Rowe's circuit wave amplitude and circuit wave phase equations are second order differential equations. However, the differential equation solver used in TWT1D solves only first order differential equations. Moreover, the equations must be of the form,

$$\begin{aligned}
y_1'(t) &= f_1\big(t,\ y_1(t),\ y_2(t),\ \ldots,\ y_n(t)\big) \\
y_2'(t) &= f_2\big(t,\ y_1(t),\ y_2(t),\ \ldots,\ y_n(t)\big) \\
\vdots\quad & \qquad\quad \vdots \qquad\qquad\qquad\qquad \vdots \\
y_n'(t) &= f_n\big(t,\ y_1(t),\ y_2(t),\ \ldots,\ y_n(t)\big)
\end{aligned} \tag{37}$$

So, the equations must be manipulated to fill these requirements. Let

$$\frac{dA_n(y)}{dy} = G_n(y) \qquad \text{and} \qquad \frac{d\theta_n(y)}{dy} = F_n(y) \tag{38}$$

then

$$\frac{d^2A_n(y)}{dy^2} = \frac{dG_n(y)}{dy} \qquad \text{and} \qquad \frac{d^2\theta_n(y)}{dy^2} = \frac{dF_n(y)}{dy} \tag{39}$$

Substituting these new variables into Rowe's equations and rearranging yields

$$\frac{\partial\Phi_n(y,\Phi_{01})}{\partial y} = 2\left(\frac{\omega_n}{\omega_1}\right)\frac{u(y,\Phi_{01})}{1 + 2C_1u(y,\Phi_{01})} - F_n(y) \tag{40}$$

$$\frac{dA_n(y)}{dy} = G_n(y) \tag{41}$$

$$\frac{dG_n(v)}{dv} = A_n(v)\left[\left(\frac{\omega_n}{\omega_1 C_1} - F_n(v)\right)^2 - \left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_n b_n}{C_1}\right)^2\right]$$

$$+ \left(\frac{Z_{0n}}{Z_{01}}\right)\left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_n b_n}{C_1 \pi}\right)\left[\int_0^{2\pi} \frac{\cos \Phi_n(v,\Phi'_{01})\, d\Phi'_{01}}{1 + 2C_1 u(v,\Phi'_{01})}\right.$$

$$\left. + 2C_n d_n \int_0^{2\pi} \frac{\sin \Phi_n(v,\Phi'_{01})\, d\Phi'_{01}}{1 + 2C_1 u(v,\Phi'_{01})}\right] \qquad (42)$$

$$\frac{d\theta_n(v)}{dv} = F_n(v) \qquad (43)$$

$$\frac{dF_n(v)}{dv} = 2C_n d_n\left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_n b_n}{C_1}\right)^2 + \frac{1}{A_n(v)}\left\{2G_n(v)\left(\frac{\omega_n}{\omega_1 C_1} - F_n(v)\right)\right.$$

$$+ \left(\frac{Z_{0n}}{Z_{01}}\right)\left(\frac{\omega_n}{\omega_1}\right)^2\left(\frac{1 + C_n b_n}{C_1 \pi}\right)\left[\int_0^{2\pi} \frac{\sin \Phi_n(v,\Phi'_{01})\, d\Phi'_{01}}{1 + 2C_1 u(v,\Phi'_{01})}\right.$$

$$\left.\left. - 2C_n d_n \int_0^{2\pi} \frac{\cos \Phi_n(v,\Phi'_{01})\, d\Phi'_{01}}{1 + 2C_1 u(v,\Phi'_{01})}\right]\right\} \qquad (44)$$

$$\frac{\partial u(v,\Phi_{01})}{\partial v} = \frac{C_1}{1 + 2C_1 u(v,\Phi_{01})} \sum_n \left[G_n(v) \cos \Phi_n - A_n(v) \sin \Phi_n\left(\frac{\omega_n}{\omega_1 C_1} - F_n(v)\right)\right]$$

$$+ \left(\frac{1}{1 + C_n b_n}\right)\left(\frac{\omega_p}{\omega_1 C_1}\right)^2 \int_0^{2\pi} \frac{f(\Phi - \Phi'_{01})\, d\Phi'_{01}}{1 + 2C_1 u(v,\Phi_{01})} \qquad (45)$$

These are the differential equations that are set up and evaluated in subroutines DIFFEQ, DIFF, DRFTEQ, and DRIFT. The subroutines also set the initial conditions for the large-signal equations.

Subroutine DIFFEQ contains Rowe's differential equations. The computer names used for the various variables are given in Table 2.

Table 2. Computer representation of wave variables.

Variable	Representation
$A_n(y)$	A(N)
$\frac{dA_n(y)}{dy}$	DA(N)
$G_n(y)$	G(N)
$\frac{dG_n(y)}{dy}$	DG(N)
$\theta_n(y)$	THETA(N)
$\frac{d\theta_n(y)}{dy}$	DTHETA(N)
$F_n(y)$	F(N)
$\frac{dF_n(y)}{dy}$	DF(N)
$\Phi_n(y,\Phi_{01})$	PHI(I,N)
$\frac{\partial\Phi_n(y,\Phi_{01})}{\partial y}$	DPHI(I,N)
$u(y,\Phi_{01})$	U(I)
$\frac{\partial u(y,\Phi_{01})}{\partial y}$	DU(I)
$\int_0^{2\pi} \frac{\sin \Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1u(y,\Phi'_{01})}$	ISIN(N)
$\int_0^{2\pi} \frac{\cos \Phi_n(y,\Phi'_{01})d\Phi'_{01}}{1 + 2C_1u(y,\Phi'_{01})}$	ICOS(N)
$\int_0^{2\pi} \frac{F(\Phi - \Phi'_{01})d\Phi'_{01}}{1 + 2C_1u(y,\Phi_{01})}$	IFUNCT(I)

This subroutine is used by the differential equations solver to evaluate the derivatives. After the equations are evaluated, they are stored in a single array called EQNS. The solutions to the equations are stored in the array, VALUES.

Subroutine DIFF sets the initial conditions for the differential equations and calls the differential equation solver. The parameters passed to the solver are the subroutine used to evaluate the derivatives, DIFFEQ, the number of equations to be integrated, NEQN; the array of solutions, VALUES; the independent variable, Y; the position where the solution is desired, YPRINT; the relative and absolute local error tolerances RELERR and ABSERR; the flag to initialize solver, IFLAG; and the flag to set the machine dependent variable, SETUP.

Subroutine DRFTEQ contains the differential equations for a drift tube. It is used by the differential equation solver to evaluate the derivatives when integrating through a sever. As in subroutine DIFFEQ, the equations are stored in a single array called EQNS, and the solutions to the equations are stored in the array, VALUES.

Subroutine DRIFT performs the same duties for DRFTEQ that DIFF performs for DIFFEQ.

C. Integrals in Rowe's Differential Equations

The circuit wave amplitude, the circuit wave phase, and the electron force equations each contain integrals that must be solved at every step of the differential equation solution. These integrals are solved in subroutines FINT, INT1, AND INT2.

Subroutine FINT solves the two integrals that arise from the Fourier series expansion of the beam charge density. Since the electron disks are evenly spaced over 2π radians initially, the integration over phase can be replaced with an integration over all the electron disks. Moreover, the discrete number of electron disks used in the computer model allows the integrals to be replaced by summations. These summations are conducted once at each step of the differential equation solution.

Subroutine INT1 solves the integral from the space-charge field expression. As in the beam charge density integrals, this integral is replaced by a summation over the electron disks. The space-charge summation must be conducted for each electron disk at each step.

Subroutine INT2 solves the polynomial approximation to the space-charge integral. This subroutine runs in about half the time that INT1 takes to run. Since the space-charge integral is evaluated so often, using the polynomial approximation can make a significant difference in run time.

D. Numerical Analysis

Numerical analysis routines are used to solve Rowe's large-signal equations and to calcualte Bessel functions for Pierce's gain parameter. The numerical analysis routines are DESOLV, STEP, INTRP, MACHIN, and BESSEL.

The differential equation solver used in TWT1D was taken from a text by L. F. Shampine and M. K. Gordon.[8] The solver is a variable step, variable order Adams code. It consists of routines DESOLV, STEP, INTRP, and MACHIN. Subroutine DESOLV is a driver that calls the other subroutines in the solver. MACHIN computes the machine unit roundoff error, U, which is the smallest positive number such that 1.0 + U > 1.0. Subroutine STEP integrates the differential equations one step, and subroutine INTRP approximates the solution at the end point by evaluating the polynomial there. The code is completely explained and documented in the text by Shampine and Gordon, and those interested in the code should consult that text.

Subroutine BESSEL calculates modified Bessel functions of the first and second kind.

E. Program Output

As Rowe's equations are integrated down the traveling-wave tube, various parameters describing the tube's performance are printed to output data and plot files and displayed on the terminal. In addition, variables describing the beam and circuit wave must be printed for a restart printout. The subroutines that handle this are OUTPUT and OUTRS.

Subroutine OUTPUT controls the output of data at each print interval. It prints the axial position, power, and phase angle to the plot file, and displays the position and power on the terminal. The NPR variable is used to determine which variables are printed to the output data file. In addition, OUTPUT uses the power and its derivative with respect to position to calculate the saturated power.

Subroutine OUTRS controls the restart printouts. A restart output is used when an engineer wants to examine the effect of different output helices without having to run the input helix in each case. The circuit wave variables printed out are normalized position, Y, circuit wave amplitude, A(n), its derivative, DA(n), circuit wave phase, THETA(n), and its derivative, DTHETA(n). The beam variables printed out are electron phase positions, PHI(i,n), and electron velocities, U(i). In addition, if the polynomial approximation to the space-charge function is used, its coefficients, CB(m) are printed out. These data contain the information needed to restart a run at that position.

F. Circuit Changes

Whenever a change in circuit velocity or loss occurs, the parameters describing the circuit must be recomputed. Subroutine ATTEN handles changes in loss due to attenuators or severs, and subroutine CHANGE handles changes in velocity due to velocity steps or tapers.

Attenuators are broken into 20 sections in the main program. In each section, subroutine ATTEN adjusts Pierce's loss parameter, calls a subroutine to integrate the large-signal equations, and checks the event code list to see if any velocity changes or restart printouts occur in the section. If the loss in a section exceeds 170 dB, the section is treated as a sever, and the drift tube equations are integrated. At the end of the sever, the subroutine sets initial conditions and integrates the large-signal equations. After all the sections are integrated, the loss parameter is reset to its original value.

Subroutine CHANGE first checks to see if the velocity change includes a change in helix radius. If so, the fill factor is rescaled, and the voltage, beam velocity, and plasma frequency are recalculated.

The phase velocity of the new helix section is used to recalculate the propagation constants, the axial impedance is used to recalculate the Pierce's gain and velocity parameters, and the circuit loss is used to recalculate Pierce's loss parameter. Using the new values, the normalized helix voltage and its derivative are rescaled, and the plasma frequency reduction factor and the space-charge parameter are recalculated. Finally, the axial position, the length of the tube, and the print interval are renormalized.

G. Diagnostics

Subroutines ENERGY and BADIN are provided for diagnostic purposes. These subroutines can help indicate whether or not the model used is reasonable, or if the input file has incorrect data.

Subroutine ENERGY checks to see if energy is conserved in the computer model. It calculates the power lost by the beam, EBEAM, and the power contained in the circuit and its fields, EWAVE. In addition, it computes an energy balance, EBLNC = $(|\mathrm{EBEAM}| - |\mathrm{EWAVE}|)/(|\mathrm{EBEAM}| + |\mathrm{EWAVE}|)$. If the model is a reasonable simulation, EBLNC will be near zero. However, this routine does not account for the circuit wave power lost in attenuators so, in these regions, EBLNC will not be near zero.

Subroutine BADIN is called when the input to subroutine DESOLV is incorrect. When called, BADIN prints an error message to the screen and halts program execution. This subroutine will be called only if the input data file has incorrect data.

H. Rowe's Space-Charge Expression

Rowe's space-charge weighting function is very complex and expensive to evaluate. In fact, the majority of the computer time used during a run is spent evaluating and integrating this function. In order to reduce the amount of time spent computing the space-charge forces, the space-charge weighting function can be represented by a series expansion. The subroutines that generate the series expansion are SPACE and SIMINT.

The expansion of a function, f(x), in a series of Chebyshev polynomials is given by

$$f(x) = \frac{a_0}{2} \sum_{i=1}^{\infty} a_i T_i(x) \qquad (46)$$

where $T_i(x)$ are the Chebyshev polynomials and

$$a_i = \frac{2}{\pi} \int_{-1}^{+1} \frac{f(x) T_i(x)}{\sqrt{1 - x^2}} \, dx \qquad (47)$$

Over the interval (0, 2π), the function becomes

$$F\left(\Phi - \Phi'_{01}\right) = \frac{a_0}{2} \sum_{i=1}^{\infty} a_i T_i(x) \qquad (48)$$

where

$$a_i = \frac{2}{\pi} \int_{-1}^{+1} \frac{F\left(\pi(x + 1)\right) T_i(x)}{\sqrt{1 - x^2}}$$

and

$$x = \frac{\Phi - \Phi'_{01}}{\pi} - 1$$

As Fig. 8 shows, Rowe's space-charge function is odd. Therefore, only the odd order terms are used for the series expansion. For a fifth order expansion, the series is

$$F(\Phi - \Phi'_{01}) \approx a_1T_1\left(\frac{\Phi - \Phi'_{01}}{\pi} - 1\right) + a_3T_3\left(\frac{\Phi - \Phi'_{01}}{\pi} - 1\right) + a_5T_5\left(\frac{\Phi - \Phi'_{01}}{\pi} - 1\right) \tag{49}$$

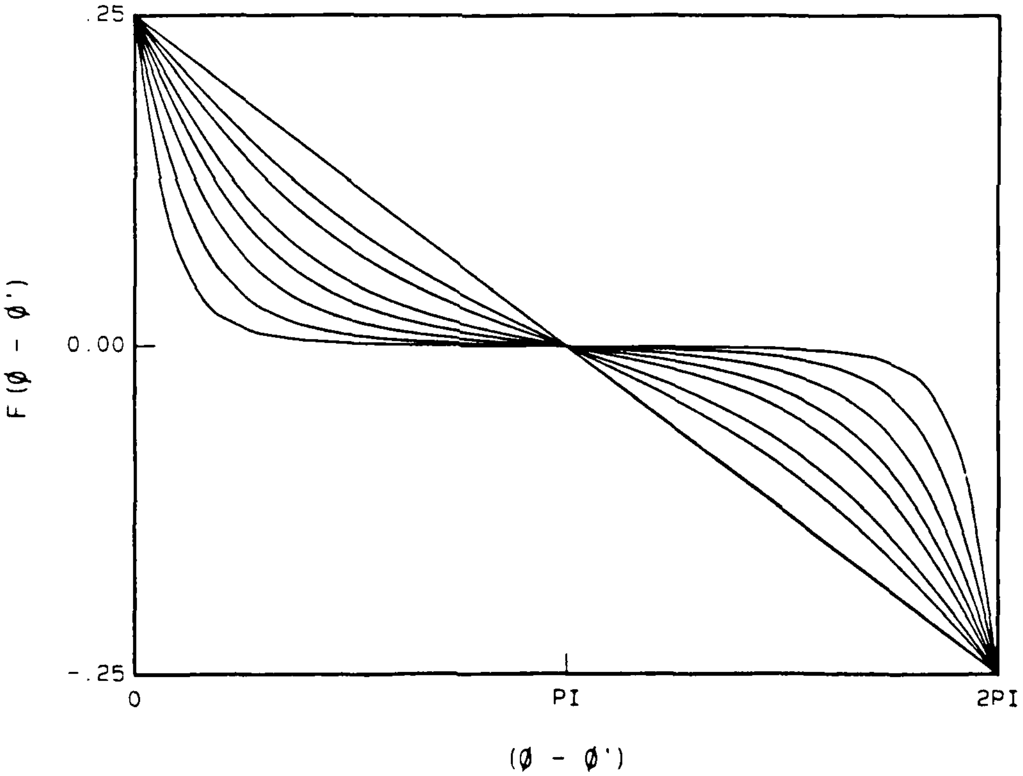


Fig. 8. Rowe's space-charge function.

where

$$a_1 = \frac{2}{\pi} \int_{-1}^{+1} \frac{F\left(\pi(x + 1)\right)x}{\sqrt{1 - x^2}}\, dx$$

$$a_3 = \frac{2}{\pi} \int_{-1}^{+1} \frac{F\left(\pi(x + 1)\right)\ \left(4x^3 - 3x\right)}{\sqrt{1 - x^2}}\, dx$$

$$a_5\ d= \frac{2}{\pi} \int_{-1}^{+1} \frac{F\left(\pi(x + 1)\right)\ \left(16x^5 - 20x^5 + 5x\right)}{\sqrt{1 - x^2}}\, dx$$

This expression is plotted along with Rowe's space-charge function for the case of f(a'/b') = 1.25 in Fig. 9. As the figure shows, the agreement between the two functions is very good. Figure 10 compares results

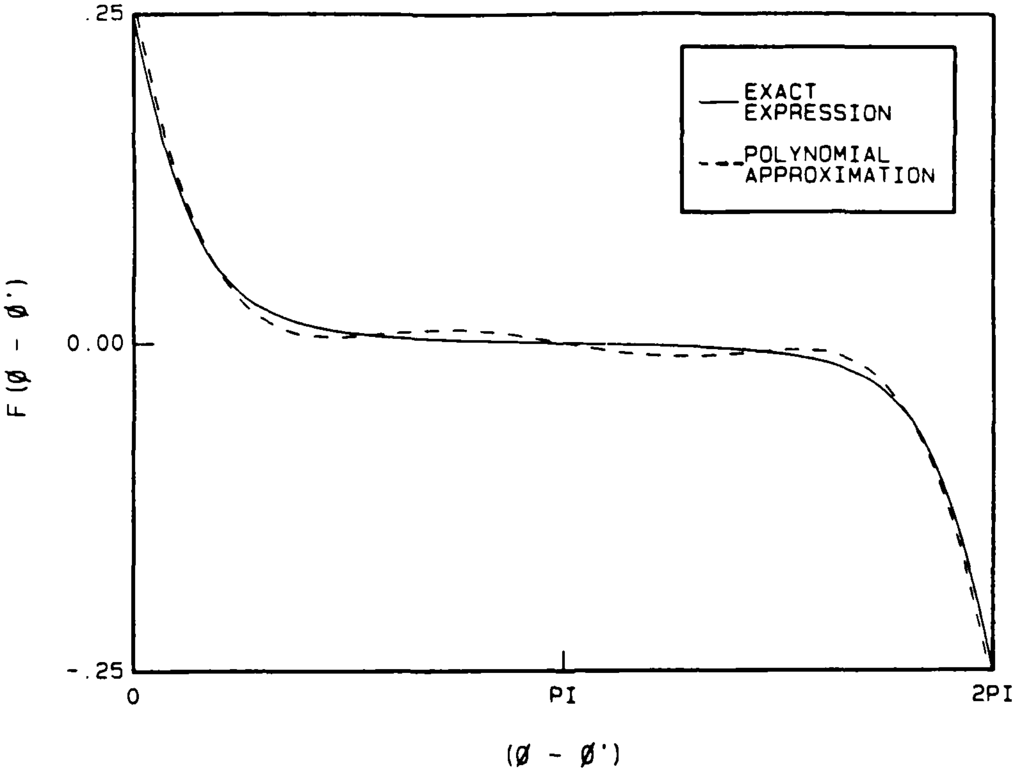


Fig. 9. Polynomial approximation to Rowe's space-charge function.

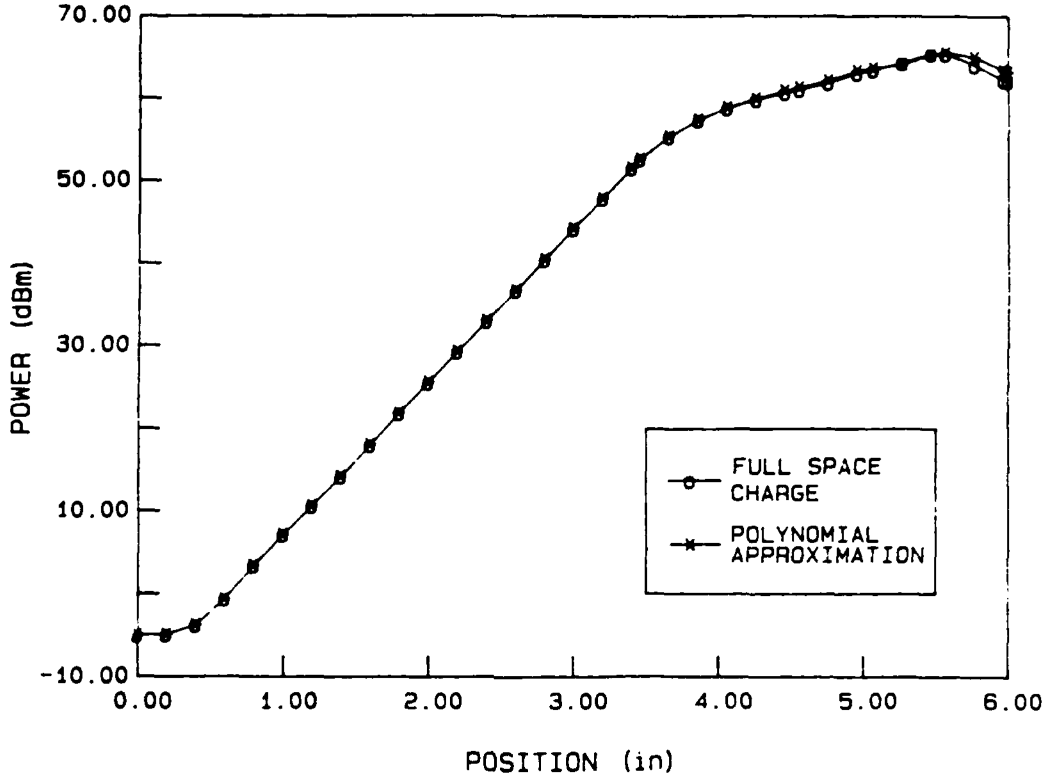


Fig. 10. Power plot using the polynomial approximation.

of a TWT1D run using the full space-charge expression to a run on the same tube using the polynomial approximation. The results of the two runs are almost identical, but the run using the approximation took only half the time of the run using Rowe's full space-charge expression.

Subroutine SPACE calculates the coefficients for the polynomial approximation using the function SIMINT. SIMINT solves the integral for the coefficients using Simpson's integration.

V. RESULTS AND CONCLUSION

In this section, the validity of TWT1D is investigated. Two Teledyne MEC tubes were simulated by the program, and the results are compared to the measured output of the tubes.

The first tube modeled is an I/J band pulse tube. It has 6 attenuators, 4 pitch changes, and a diameter step. The tube was simulated, using 16 electron disks and Rowe's full space-charge expression. Several runs of the program were made to determine the saturated power across the band. The results of the program and the measured output power are plotted in Fig. 11. As the figure shows, the agreement across the band is very good. However, at the low end of the band, the predicted power is 1 to 2 dB higher than the measured output. In order to correct this, the program was rerun for the first three data points with the harmonic included. Since the tube is a broadband device, harmonics can have a significant effect on the output power at the low end of the band. The results of the runs with the harmonic included are shown in Fig. 12. As this figure shows, the harmonic does lower the saturated power and brings the 2 curves into closer agreement at the lower frequencies.

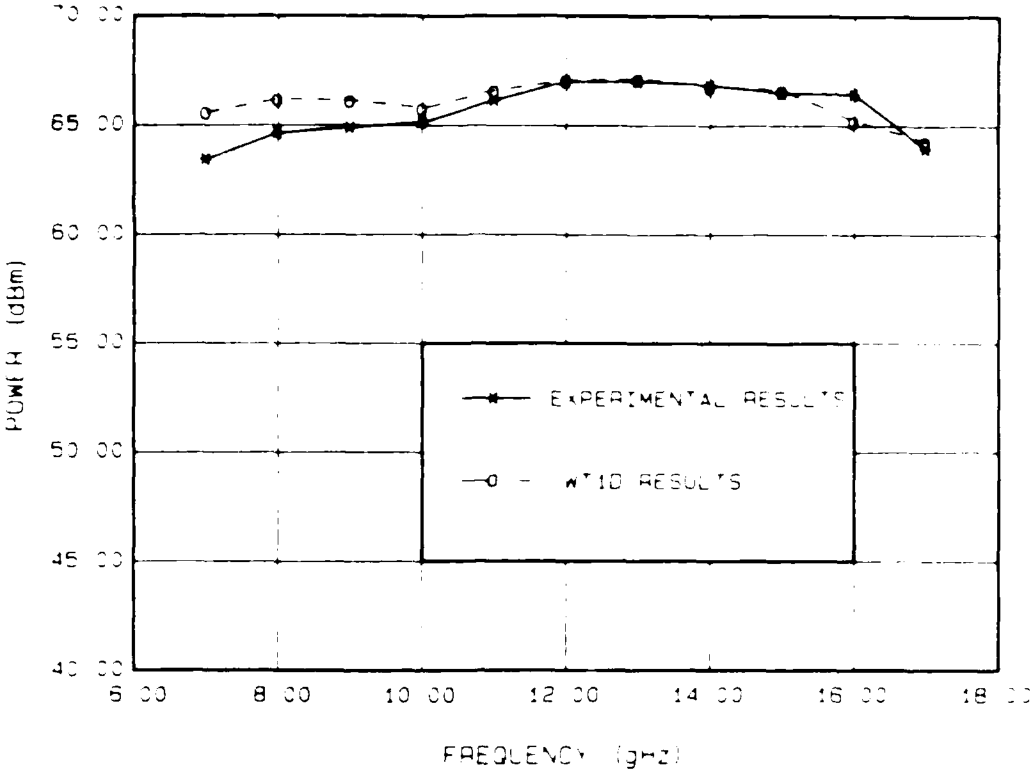


Fig. 11. Saturated power for an I/J band tube.

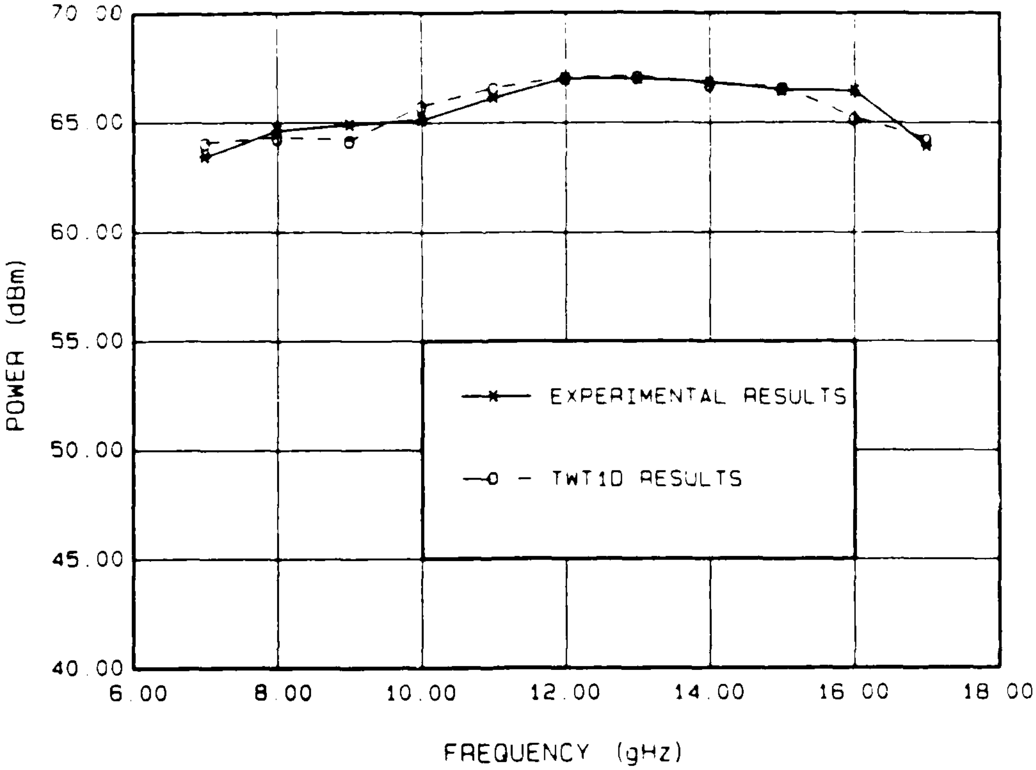


Fig. 12. Saturated power for an I/J band tube with harmonic.

The second tube modeled is an E/H band CW tube. It has 3 attenuators and 2 pitch changes. The tube was simulated, using 16 electron disks and the polynomial approximation to the space-charge expression. The results of the runs and the measured output are plotted in Fig. 13. Again, the agreement between the two curves is very good.

The simulations of the two tubes indicate that TWT1D can be a useful tool for the design engineer. However, care must be taken whenever a nonlinear program is used. Many of the parameters that are used by such a program, such as the fill factor and the axial impedance, are not precisely known to the engineer. Yet these parameters have a large effect on the output of the program. Figure 14 shows the results that changing the fill factor can have on the output of the program. Note that changing the fill factor from 0.4 to 0.6 moves the saturation position as much as 2 inches.

Perhaps the biggest problem with any large-signal program based on Rowe's approach is the way it models attenuators. As was discussed in Section III, attenuators cause oscillations in the circuit wave phase angle. This problem was circumvented in TWT1D by replacing attenuators with severs, but that is not a complete solution. This problem should be solved before more elaborate models of Rowe's approach are attempted.

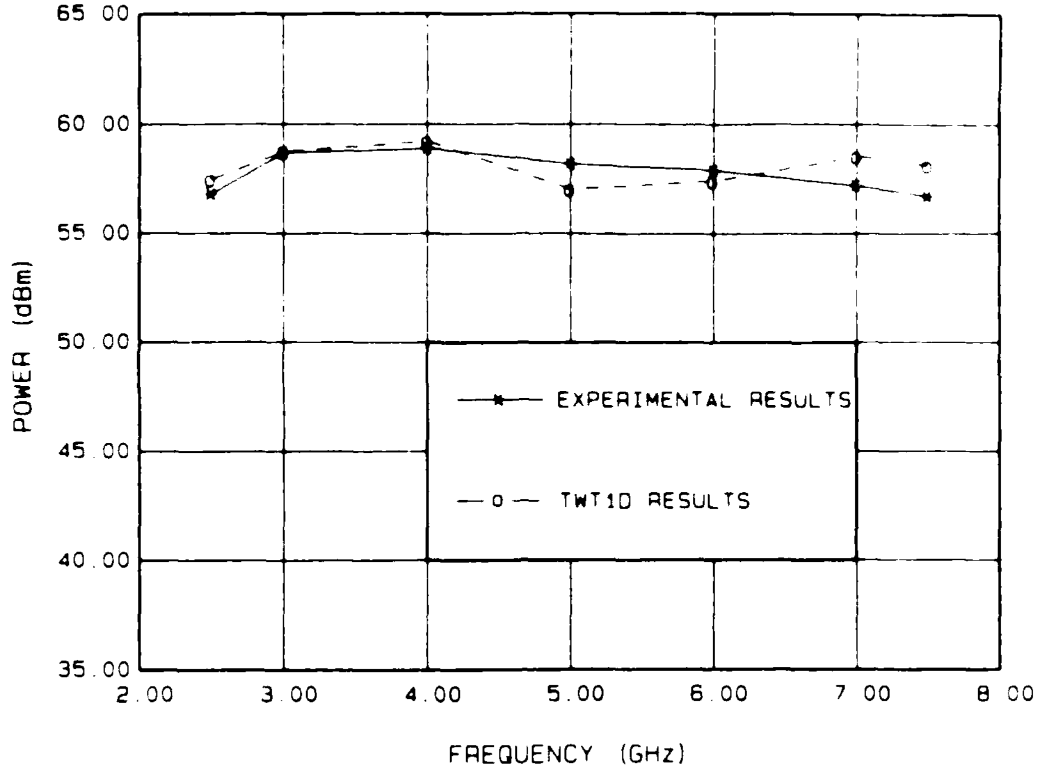


Fig. 13. Saturated power for an E/H band tube.

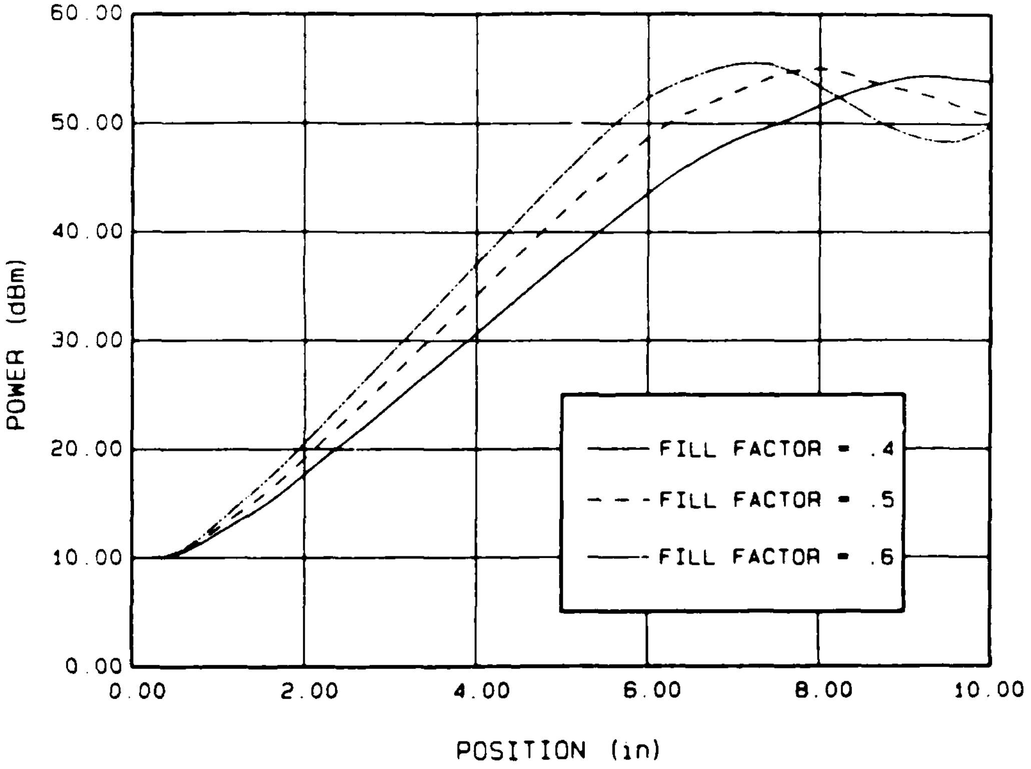


Fig. 14. Effect of fill factor on saturation.

REFERENCES

1. M. K. Scherba and J. E. Rowe, "Characteristics of Multisignal and Noise-Modulated High-Power Microwave Amplifiers," IRE Transactions on Electron Devices, Vol. ED-18, January 1971, pp. 11-80.

2. A. Norsieck, "Theory of Large-Signal Behavior of Traveling-Wave Amplifiers," Proceedings of the IRE, Vol. 41, May 1953, pp. 630-637.

3. H. C. Poulter, "Large-Signal Theory of the Traveling-Wave Tube," Electronics Research Laboratory Technical Report No. 73, Stanford University, Stanford, California, January 1954.

4. J. E. Rowe, "A Large-Signal Analysis of the Traveling-Wave Amplifier: Theory and Results," IRE Transactions on Electron Devices, Vol. ED-3, January 1956, pp. 39-56.

5. L. Brillouin, "The Traveling-Wave Tube (Discussion of Waves for Large Amplitudes)," Journal of Applied Physics, Vol. 20, December 1949, pp. 1196-1206.

6. J. R. Pierce, Traveling-Wave Tubes, D. Van Nostrand Company, New York, 1950.

7. J. E. Rowe, Nonlinear Electron-Wave Interaction Phenomenon, Academic Press, Inc., New York, 1960.

8. L. F. Shampine and M. K. Gordon, Computer Solution of Ordinary Differential Equations; The Initial Value Problem, W. H. Freeman & Company, San Francisco, California, 1975.

9. R. Baer, "The WARTHOG Large-Signal Program," Technical Memorandum 143, Teledyne MEC, Palo Alto, California, June 1980.

APPENDIX A

TWTID SOURCE CODE LISTING

```
FTN77,S
$CDS ON
$FILES(3,3)
$EMA //
$EMA /TEMP/
      PROGRAM TWTID(3,91)
C     LARGE SIGNAL ANALYSIS PROGRAM FOR HELIX TWT'S
C     BASED ON THEORY BY J.E. ROWE & M.K. SHERBA
C     BY THERESA BRUNASSO
C     TELEDYNE MEC
C     3165 PORTER DRIVE
C     PALO ALTO, CA 94304
C     LAST REVISED  <860515.0717>
C     -------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      CHARACTER*64 IFILE,OFILE,PFILE
      INTEGER ATYPE(6),RSTART,SETU,SCS,SWITCH,SWLIST,TODAY(15),
     &TRIMLEN,TYPE
      DOUBLE PRECISION AISTEP(3),AITEMP(6,3),CRTEMP(6),FI(3),ICOS,
     &IFUNCT,ISIN,IO,IOGB,I1GB,K1,K2,K3,KOGB,K1GB,LA,LAMAX(6),LT,MAG,
     &NORMZ,RFSTEP(3),RFTEMP(6,3),SHAPE(20,4),TAPER(6),VPSTEP(3),
     &VPTEMP(6,3),ZA(19,6),ZATNE(6),ZATNS(6),ZC(60),ZI(6),ZLIST(60),
     &ZRST(10)
      COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),A0(3),B(3),BETA(3),
     &BETA0(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
     &DF(3),EBEAM,EBLNC,EM,EPSI,ESAT.ETA,EWAVE,EXP1,F(3),FILL,G(3),
     &GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),IO,K1,K2,K3,
     &LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
     &PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
     &RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
     &THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,V0,W(3),WP,WPWC,
     &YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
      COMMON /TEMP/ FTEMP(64,64)
      DATA SHAPE/ .023,.033,.05,.067,.092,.12,.15,.193,.257,.333,.46,
     &.593,.733,.9,1.,.9,.733,.5,.233,.033,.013,.030,.066,.129,.231,
     &.375,.553,.739,.897,.990,.990,.897,.739,.553,.375,.231,.129,.066,
     &.03,.013,.033,.233,.5,.733,.90,1.,.90,.733,.593,.46,.333,.257,
     &.193,.15,.12,.092,.067,.05,.033,.023,1.,1.,1.,1.,1.,1.,1.,1.,1.,
     &1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,1./
      CALL FPARM(IFILE,OFILE,PFILE)
      LU=1
      PI=4.*ATAN(1.)
      EM=1.7588047D11                 'EM=e/m (Coul/kg)!
      EPSI=8.85418782D-12             !PERMITIVITTY OF FREE SPACE (F/m)'
      K1=.5/(PI*EPSI*SQRT(2.*EM))     !K1=1/(2*PI*EPSI*SQRT(2*e/m))'
      K2=.5/EM*2.99792458D08**2       'K2=(m*c**2)/(2*e)!
      K3=.5/K2
      SMALLC=2.99792458D08*39.37      'SPEED OF LIGHT (in/s)'
      NCHNG=0
      NATTN=0
      SETU=0
      FIRST=0
      ND=0
      DO I=1,64
         FTEMP(I,I)=0
      END DO
C=================================================================
C     OPENING DISPLAY
C     ---------------
      CALL FTIME(TODAY)
      WRITE(LU,10)(TODAY(I),I=1,15)
   10 FORMAT(//15X,'TWTID     ',15A2/)
      IF (IFILE(1:1) .EQ. CHAR(0)) THEN
         WRITE(LU,20)
   20    FORMAT(//' INPUT DATA FILE NAME? _')
         READ(LU,30)IFILE
   30    FORMAT(A)
      END IF
      IF (OFILE(1:1) .EQ. CHAR(0)) THEN
         WRITE(LU,40)
   40    FORMAT(//' OUTPUT FILE NAME? _')
         READ(LU,30)OFILE
      END IF
      IF (PFILE(1:1) .EQ. CHAR(0)) THEN
         WRITE(LU,50)
   50    FORMAT(//' POWER PLOT FILE NAME? _')
         READ(LU,30)PFILE
      END IF
      WRITE(LU,60)OFILE(1:TRIMLEN(OFILE)),PFILE(1:TRIMLEN(PFILE))
   60 FORMAT(/' ALL OUTPUT DATA WILL BE ROUTED TO FILE ',A/
     &' ALL PLOT DATA WILL BE ROUTED TO FILE ',A)
C======================================================================
C     READ DATA FROM INPUT FILE
C     -------------------------
      OPEN(99,FILE=IFILE)
      READ(99,*)V0,I0,RAD,FILL,TPI,ZF
      READ(99,*)NUMF,NUMC,NUMA,NUMR
      DO I=1,NUMF
         READ(99,*)FI(I),PINI(I),PHVEL(I),AXIMP(I),RFLSS(I),THETA(I)
      END DO
      IF (NUMC .GT. 0) THEN
         DO I=1,NUMC
            READ(99,*)ZI(I),CRTEMP(I),TAPER(I)
         END DO
         READ(99,*)((VPTEMP(I,J),I=1,NUMC),J=1,NUMF)
         READ(99,*)((AITEMP(I,J),I=1,NUMC),J=1,NUMF)
         READ(99,*)((RFTEMP(I,J),I=1,NUMC),J=1,NUMF)
      END IF
      IF (NUMA .GT. 0) THEN
         DO I=1,NUMA
            READ(99,*)ZATNS(I),ZATNE(I),LAMAX(I),ATYPE(I)
         END DO
         DO I=1,NUMA              !SET UP ATTENUATOR SECTIONS'
            ASTEP=(ZATNE(I)-ZATNS(I))*.05
            ZA(1,I)=ZATNS(I)+ASTEP
            LA(1,I)=LAMAX(I)*SHAPE(1,ATYPE(I))
            DO K=2,19
               LA(K,I)=LAMAX(I)*SHAPE(K,ATYPE(I))
               ZA(K,I)=ZA(K-1,I)+ASTEP
            END DO
            LA(20,I)=LAMAX(I)*SHAPE(20,ATYPE(I))
         END DO
      END IF
      NUMT=0
      K=0
      IF (NUMC .GT. 0) THEN                 !SET UP TAPERS!
```

```
         DO I=1,NUMC
            IM1=I-1
            IF (TAPER(I) .EQ. 0) THEN         'PITCH CHANGE'
               K=K+1
               NC=NUMT*10+K
               ZC(NC)=ZI(I)
               CKTRAD(NC)=CRTEMP(I)
               DO J=1,NUMF
                  PHVELN(NC,J)=VPTEMP(I,J)
                  AXIMPN(NC,J)=AITEMP(I,J)
                  RFLSSN(NC,J)=RFTEMP(I,J)
               END DO
            ELSE                              !VELOCITY TAPER'
               NUMT=NUMT+1
               TSTEP=TAPER(I)*.1
               NT=1+10*(NUMT-1)+K
               ZC(NT)=ZI(I)
               IF (I .EQ. 1) THEN       !FIRST PITCH CHANGE'
                  DO J=1,NUMF
                     VPSTEP(J)=(VPTEMP(1,J)-PHVEL(J))*.1
                     AISTEP(J)=(AITEMP(1,J)-AXIMP(J))*.1
                     RFSTEP(J)=(RFTEMP(1,J)-RFLSS(J))*.1
                     PHVELN(NT,J)=PHVEL(J)+VPSTEP(J)
                     AXIMPN(NT,J)=AXIMP(J)+AISTEP(J)
                     RFLSSN(NT,J)=RFLSS(J)+RFSTEP(J)
                  END DO
               ELSE
                  DO J=1,NUMF
                     VPSTEP(J)=(VPTEMP(I,J)-VPTEMP(IM1,J))*.1
                     AISTEP(J)=(AITEMP(I,J)-AITEMP(IM1,J))*.1
                     RFSTEP(J)=(RFTEMP(I,J)-RFTEMP(IM1,J))*.1
                     PHVELN(NT,J)=VPTEMP(IM1,J)+VPSTEP(J)
                     AXIMPN(NT,J)=AITEMP(IM1,J)+AISTEP(J)
                     RFLSSN(NT,J)=RFTEMP(IM1,J)+RFSTEP(J)
                  END DO
               END IF
               CKTRAD(NT)=CRTEMP(I)
               DO NSTEP=2,10
                  DO J=1,NUMF
                     NT=10*(NUMT-1)+NSTEP+K
                     NTM1=NT-1
                     ZC(NT)=ZI(I)+(NSTEP-1)*TSTEP
                     CKTRAD(NT)=CRTEMP(I)
                     PHVELN(NT,J)=PHVELN(NTM1,J)+VPSTEP(J)
                     AXIMPN(NT,J)=AXIMPN(NTM1,J)+AISTEP(J)
                     RFLSSN(NT,J)=RFLSSN(NTM1,J)+RFSTEP(J)
                  END DO
               END DO
            END IF
         END DO
      END IF
      IF (NUMR .GT. 0) READ(99,*)(ZRST(I),I=1,NUMR)
      READ(99,*)SCS,NELECT,DPC,NPR,INTSKP,RSTART,FDUMMY
      IF (RSTART .EQ. 0) CLOSE(99)
C=============================================================================
C     COMPUTE ROWE VARIABLES AND PIERCE PARAMETERS
C     ---------------------------------------------
      V2=V0*(1.-I0*V0**(-1.5)*K1*(.5-LOG(FILL)))
      VREL=K2*(1.-1./(1.+K3*V2)**2)               !RELATIVISTIC VOLTAGE!
      UZERO=SQRT(2.*EM*VREL)*39.37               !BEAM VELOCITY (in/s)!
      SMALLA=RAD                                 !HELIX RADIUS (in)!
      BP=FILL*SMALLA                             !BEAM RADIUS (in)!
      DENOM=PI*EPSI*UZERO*BP*BP
      WP=SQRT(EM*I0*61024./DENOM)                !PLASMA FREQ (rads/s)!
      DO N=1,NUMF
         VP(N)=PHVEL(N)*SMALLC                   !PHASE VELOCITY (in/s)!
         W(N)=2.*PI*FI(N)*1.D09                  !FREQUENCY (rads/s)!
         BETA(N)=W(N)/VP(N)                      !PROPAGATION CONST.!
         BETA0(N)=W(N)/SMALLC
         GAMA(N)=SQRT(BETA(N)**2-BETA0(N)**2)    !RADIAL PROP. CONST.!
         CALL BESSEL(I0GB,I1GB,K0GB,K1GB,GAMA(N)*BP)
         ZAVG=AXIMP(N)*(I0GB**2-I1GB**2)         !AVERAGED IMPEDANCE!
         C(N)=(I0*ZAVG*.25/VREL)**(1./3.)        !GAIN PARAMETER!
         B(N)=(UZERO-VP(N))/(C(N)*VP(N))         !VELOCITY PARAMETER!
         LT=RFLSS(N)/LOG10(EXP(1.))
         D(N)=LT*.05/(W(N)/UZERO*C(N))           !LOSS PARAMETER!
         A(N)=SQRT((10.**(PINI(N)*.1-3.))/(2.*C(1)*I0*VREL))
         A0(N)=A(N)
         THETA(N)=THETA(N)*PI/180.              !INJECTION ANGLE (rads)!
      END DO
      AB=SMALLA*BETA(1)
      BB=BP*BETA(1)
      W1C1=W(1)*C(1)
      WPWC=WP/W1C1
      IF (SCS .GT. 0) THEN      !COMPUTE SPACE CHARGE REDUCTION FACTOR!
         CALL SPACE(AB,BB,SCS,R,EXP1,CB,PI)
      ELSE                      !SPACE CHARGE EQUALS ZERO!
         WPWC=0.
      END IF
      WQ=R*WP                   !REDUCED PLASMA FREQUENCY!
      NORMZ=W1C1/UZERO
      QC=.25/C(1)**2*(WQ/(W(1)+WQ))**2           !SPACE CHARGE PARAMETER'
      YFINAL=ZF*NORMZ
      PRNT=DPC*NORMZ
      PSAT=PINI(1)
C==========================================================================
C     INITIALIZE CIRCUIT WAVE, OR USE RESTART DATA
C     ------------------------------------------
      IF (RSTART .EQ. 1) THEN     !READ RESTART DATA!
         READ(99,*)Y
         DO I=1,NUMF
             READ(99,*)A(I),G(I),THETA(I),F(I)
         END DO
         DO I=1,NELECT
             READ(99,*)(PHI(I,J),J=1,NUMF)
         END DO
         READ(99,*)(U(I),I=1,NELECT)
         IF (SCS .EQ. 2) READ(99,*)(CB(I),I=1,5)
         CLOSE(99)
      ELSE                         !INITIALIZE BEAM & CIRCUIT WAVE!
         Y=0
         DO I=1,NELECT
            U(I)=0
         END DO
         IF (FDUMMY .EQ. 0) THEN
            DO N=1,NUMF
               RN=FLOAT(N)
               DO I=1,NELECT
```

```
                  RI=FLOAT(I)
                  PHI(I,N)=2.*RN*PI*(RI-1.)/FLOAT(NELECT)+(RN*THETA(1)
     &                         -THETA(N))
               END DO
            END DO
         ELSE
            DO N=1,NUMF
               M=FI(N)/FDUMMY
               DO I=1,NELECT
                  PHI(I,N)=2*M*PI*(I-1)/NELECT-THETA(N)
               END DO
            END DO
         END IF
         DO N=1,NUMF                   !INITIAL COND. FOR CIRCUIT WAVE!
            G(N)=-D(N)*A(N)*(1.+C(N)*B(N))*W(N)*C(N)/W1C1
            F(N)=-B(N)*W(N)*C(N)/W1C1
         END DO
      END IF
      PW=2.*C(1)*I0*VREL*A(1)*A(1)*(1.-C(1)*F(1))/(1+C(1)*B(1))
      OLDPDB=10*LOG10(PW)+30
      OLDZ=Y/NORMZ
      SKPINT=0
C========================================================================
C     PRINT OUT VALUES CALCULATED FROM INPUT
C     ---------------------------------------
      OPEN(98,FILE=OFILE)
      OPEN(97,FILE=PFILE)
      IF (NPR .EQ. 4) OPEN(96,FILE='ELECTRON.PLT')
      WRITE(97,3)OLDZ,(PINI(N),N=1,NUMF),(THETA(M),M=1,NUMF)
   3  FORMAT(1X,F7.4,6(2X,F8.4))
      CALL FTIME(TODAY)
      WRITE(98,10)(TODAY(I),I=1,15)
      WRITE(98,5)IFILE(1:TRIMLEN(IFILE))
    5 FORMAT(20X,'RESULTS FOR INPUT FILE ',A//)
      WRITE(98,15)
   15 FORMAT(' N  FREQ.(gHz)  INPUT PWR.(dBm)  REL. PHASE(rad)')
      DO N=1,NUMF
         WRITE(98,25)N,FI(N),PINI(N),THETA(N)
   25    FORMAT(1X,I1,3X,F8.4,7X,F8.4,10X,F8.4)
      END DO
      WRITE(98,35)I0,V0,VREL,UZERO/SMALLC,TPI,WP,R
   35 FORMAT(/15X,'CATHODE CURRENT(Amps)= ',F7.4/16X,'HELIX VOLTAGE(Volt
     &s)= ',F9.2/12X,'EFFECTIVE VOLTAGE(Volts)= ',F9.2/12X,'NORMALIZED B
     &EAM VELOCITY= ',F7.4/20X,'HELIX PITCH(tpi)= ',F6.3/12X,'PLASMA FRE
     &QUENCY(rads/s)= ',E11.4/3X,'PLASMA FREQUENCY REDUCTION FACTOR= ',F
     &7.4)
      WRITE(98,45)QC,SMALLA,FILL,DPC,NELECT,NUMA,NUMC
   45 FORMAT(14X,'SPACE CHARGE PARAMETER= ',F7.4/15X,'MEAN HELIX RADIUS(
     &in)= ',F8.5/20X,'BEAM FILL FACTOR= ',F7.4/10X,'CIRCUIT PRINT INTER
     &VAL(in)= ',F7.4/12X,'NUMBER OF ELECTRON DISKS= ',I2/15X,'NUMBER OF
     & ATTENUATORS= ',I2/19X,'NUMBER OF CHANGES= ',I2/)
      WRITE(98,55)
   55 FORMAT(' N  NORM. PHASE VEL.(Vp/c)  AXIAL IMP.(ohms)  GAIN PARAMET
     &ER')
      DO N=1,NUMF
         WRITE(98,65)N,VP(N)/SMALLC,AXIMP(N),C(N)
   65    FORMAT(1X,I1,6X,F8.4,13X,F8.4,13X,F5.4)
      END DO
C========================================================================
C     SORT ALL EVENTS INTO AXIAL POSITION LIST
C     ----------------------------------------
      DO I=1,140
         ZSTOP(I)=0
         SWLIST(I)=1
      END DO
      NSTOP=0
      IF (NUMR .GT. 0) THEN        !SORT RESTART DATA POSITIONS!
         DO I=1,NUMR
            ZLIST(I)=ZRST(I)
         END DO
         NLIST=NUMR
         TYPE=2
         CALL ZSORT(TYPE,ZLIST,NLIST,ZSTOP,NSTOP,SWLIST)
      END IF
      IF (NUMA .GT. 0) THEN         !SORT ATTENUATOR START POSITIONS!
         DO I=1,NUMA
            ZLIST(I)=ZATNS(I)
         END DO
         NLIST=NUMA
         TYPE=3
         CALL ZSORT(TYPE,ZLIST,NLIST,ZSTOP,NSTOP,SWLIST)
         DO I=1,NUMA                !SORT ATTENUATOR STOP POSITIONS!
            ZLIST(I)=ZATNE(I)
         END DO
         TYPE=5
         CALL ZSORT(TYPE,ZLIST,NLIST,ZSTOP,NSTOP,SWLIST)
         DO I=1,NUMA                !SORT ATTENUATOR SECTIONS!
            DO J=1,19
               ZLIST(J)=ZA(J,I)
            END DO
            NLIST=19
            TYPE=1
            CALL ZSORT(TYPE,ZLIST,NLIST,ZSTOP,NSTOP,SWLIST)
         END DO
      END IF
      IF (NUMC .GT. 0) THEN         !SORT CIRCUIT CHANGES!
         NLIST=NUMC+9*NUMT
         DO I=1,NLIST
            ZLIST(I)=ZC(I)
         END DO
         TYPE=7
         CALL ZSORT(TYPE,ZLIST,NLIST,ZSTOP,NSTOP,SWLIST)
      END IF
      IF (RSTART .EQ. 1) THEN       !FIND RESTART POSITION!
         DO N=1,NSTOP
            SWITCH=SWLIST(N)
            CALL EVENT(SWITCH,ICHNG,IRSTRT,IASTRT,IASTOP)
            IF (IRSTRT .EQ. 1) ND=N
         END DO
      END IF
C===========================================================================
C     PROGRAM CONTROL CODE
C     --------------------
      DO WHILE (ND .LE. NSTOP)
         ND=ND+1
         YHALT=ZSTOP(ND)*NORMZ
         CALL INTSUB(Y,FIRST)
         SWITCH=SWLIST(ND)
         CALL EVENT(SWITCH,ICHNG,IRSTRT,IASTRT,IASTOP)
         IF (IRSTRT .EQ. 1) CALL OUTRS(Y)
         IF (ICHNG .EQ. 1) CALL CHANGE(Y)
         IF (IASTRT .EQ. 1) CALL ATTEN(Y,FIRST,IASTOP)
      END DO
      YHALT=ZF*NORMZ
      CALL INTSUB(Y,FIRST)
      SSGAIN=PDB1-PINI(1)
C========================================================================
C     ENDING SUMMARY
C     --------------
      IF (PDB1 .GT. PSAT) THEN      !TUBE HAS NOT SATURATED!
         PSAT=PDB1
         GSAT=PDB1-PINI(1)
         ESAT=ETA*100.
         ZSAT=Z
      END IF
      WRITE(98,70)PSAT,GSAT,ESAT,ZSAT,SSGAIN
   70 FORMAT(/26X,'ENDING PRINT OUT'/19X,'SATURATED POWER (dBm)= ',F7.3/
     &17X,'GAIN AT SATURATION (dB)= ',F7.3/15X,'SATURATION EFFICIENCY (%
     &)= ',F7.3/15X,'LENGTH AT SATURATION (in)= ',F7.3/18X,'SMALL SIGNAL
     & GAIN (dB)= ',F7.3)
      CALL FTIME(TODAY)
      WRITE(98,10)(TODAY(I),I=1,15)
      CLOSE(98)
      CLOSE(97)
      STOP
      END
C========================================================================
      SUBROUTINE SPACE(AB,BB,SCS,R,EXP1,CB,PI)
C     COMPUTES THE SPACE CHARGE REDUCTION FACTOR (R), THE
C     DERIVATIVE OF THE LOG(1-R), AND THE COEFFICIENTS (CB) FOR
C     THE POLYNOMIAL APPROXIMATION TO THE SPACE CHARGE FUNCTION
C     ---------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SCS
      DOUBLE PRECISION I0AB,I1AB,K0AB,K1AB,I0BB,I1BB,K0BB,K1BB,CA(5),
     &CB(5)
      EMA SCS,EXP1,CB,PI
      CALL BESSEL(I0AB,I1AB,K0AB,K1AB,AB)
      CALL BESSEL(I0BB,I1BB,K0BB,K1BB,BB)
      R0=SQRT(ABS(1-BB*(K0AB*I1BB+I0AB*K1BB)/I0AB))
      F0=LOG(1-R0)
      DELTA=.001
      BB=BB+DELTA
      CALL BESSEL(I0BB,I1BB,K0BB,K1BB,BB)
      R=SQRT(ABS(1-BB*(K0AB*I1BB+I0AB*K1BB)/I0AB))
      F1=LOG(1-R)
      BB=BB+DELTA
      CALL BESSEL(I0BB,I1BB,K0BB,K1BB,BB)
      R=SQRT(ABS(1-BB*(K0AB*I1BB+I0AB*K1BB)/I0AB))
      F2=LOG(1-R)
      F=-(-F2+4*F1-3*F0)/(2*DELTA)        !CENTRAL DIFFERENCE FORMULA!
      R=R0
      BB=BB-2*DELTA
      EXP1=EXP(BB*F)
      IF (SCS .EQ. 2) THEN                !POLYNOMIAL APPROXIMATION!
         DO I=1,5
            CA(I)=0
            CB(I)=0
         END DO
         DO I=1,3
            N=2*I-1
            CA(I)=2./PI*SIMINT(N,EXP1)
            CA(I)=CA(I)-SQRT(.002)/PI
         END DO
         CB(1)=CA(1)-3.*CA(2)+5.*CA(3)-7.*CA(4)+9.*CA(5)
         CB(2)=4.*CA(2)-20.*CA(3)+56.*CA(4)-120.*CA(5)
         CB(3)=16.*CA(3)-112.*CA(4)+432.*CA(5)
         CB(4)=64.*CA(4)-576.*CA(5)
         CB(5)=256.*CA(5)
      END IF
      RETURN
      END
C=========================================================================
      FUNCTION SIMINT(N,EXP1)
C     USES SIMPSON INTEGRATION TO COMPUTE THE COEFFICIENTS FOR THE
C     POLYNOMIAL APPROXIMATION TO THE SPACE CHARGE FUNCTION
C     ------------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION LOWLIM
      EMA EXP1
      SPCH(Y,EXP1)=.5/PI*(.5*(PI-Y)-2.*ATAN(SIN(Y)/(EXP1-COS(Y)))
     &   +ATAN(SIN(Y)/(EXP1**2-COS(Y)))) !ROWE'S SPACE CHARGE FUNCTION!
      CHEV(N,X)=COS(N*ACOS(X))           !CHEBYSHEV POLYNOMIALS!
      F(X)=SPCH(PI*(X+1.),EXP1)*CHEV(N,X)/SQRT(1.-X*X)  !INTEGRAND!
      PI=4.*ATAN(1.)
      DELTA=1.D-04
      LOWLIM=-1.+DELTA
      UPLIM=1.-DELTA
      SUM2=0
      SUM4=0
      X=LOWLIM+DELTA
      DO WHILE (X .LT. UPLIM)
         SUM4=SUM4+F(X)
         SUM2=SUM2+F(X+DELTA)
         X=X+2.*DELTA
      END DO
      SIMINT=DELTA/3.*(4.*SUM4+2.*SUM2+F(LOWLIM)+F(UPLIM)
     &             +4.*F(UPLIM-DELTA))
      RETURN
      END
C=========================================================================
      SUBROUTINE BESSEL(A,B,C,D,X)
C     RETURNS MODIFIED BESSEL FUNCTIONS OF 1ST AND 2ND KIND
C     A=I0, B=I1, C=K0, D=K1, X=ARGUMENT
C     -----------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      IF (X .LT. 1.0D-10) X=1.0D-10
      A=1.
      B=1.
      C=0
      G1=1.
      G2=1.
      G3=0
      G4=0
      G5=.5772157+LOG(X*.50)
      G6=X*X*.25
```

```
 10   G4=G4+1.
      G3=G3+1./G4
      G1=G1*G6/(G4*G4)
      G2=G2*G6/(G4*(G4+1.))
      A=A+G1
      B=B+G2
      C=C+G1*G3
      IF (G1*G3 .GT. 1.0D-06) GO TO 10
      C=C-A*G5
      B=.5*B*X
      IF (X .GE. 2.) THEN
         X1=2./X
         CK=X1*5.3208D-04 - 2.5154D-03
         CK=X1*CK+5.87872D-03
         CK=X1*CK-1.062446D-02
         CK=X1*CK+2.189568D-02
         CK=X1*CK-7.832358D-02
         CK=X1*CK+1.25331414
         C=CK/(SQRT(X)*EXP(X))
      END IF
      D=(1./X-B*C)/A
      RETURN
      END
C=========================================================================
$EMA //
      SUBROUTINE INTSUB(Y,FIRST)
C     INSURES THAT THE APPROPIATE PRINTOUTS ARE CONDUCTED,
C     AND THAT THE INTEGRATION GOES TO THE END OF THE INTERVAL
C     ---------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SCS,SWITCH,SWLIST,SETU
      DOUBLE PRECISION ICOS,IFUNCT,ISIN,IO,K1,K2,K3,LA,MAG,NORMZ
      COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),AO(3),B(3),BETA(3),
     &BETAO(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
     &DF(3),EBEAM,EBLNC,EM,EPSI,ESAT,ETA,EWAVE,EXP1,F(3),FILL,G(3),
     &GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),IO,K1,K2,K3,
     &LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
     &PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
     &RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
     &THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,VO,W(3),WP,WPWC,
     &YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
      IF (YHALT .GT. YFINAL) YHALT=YFINAL
      DO YPRINT=Y+PRNT,YHALT,PRNT
         CALL DIFF(Y)
         Z=Y/NORMZ
         CALL OUTPUT(FIRST)
      END DO
      IF (Y .LT. YHALT) THEN
         YPRINT=YHALT
         CALL DIFF(Y)
         Z=Y/NORMZ
         CALL OUTPUT(FIRST)
      END IF
      RETURN
      END
C=========================================================================
      SUBROUTINE ZSORT(TYPE,ZLIST,NLIST,ZSTOP,NSTOP,SWLIST)
C     SORTS THE POSITIONS AT WHICH CIRCUIT CHANGES, ATTENUATORS,
C     ETC. OCCUR, AND FORMS THEM INTO A SINGLE LIST.
```

```
C           ZLIST=LIST OF STOP POSITIONS, NLIST=LENGTH OF ZLIST,
C           ZSTOP=SORTED LIST OF STOP POSITIONS, NSTOP=LENGTH OF ZSTOP
C           SWLIST=LIST OF EVENT TYPES CORRESPONDING TO ZSTOP
C        --------------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SWLIST,TYPE
      DIMENSION ZLIST(NLIST),ZSTOP(140),SWLIST(140)
      EMA ZSTOP,SWLIST
      DO I=1,NLIST
         N=1
         DO WHILE (ZLIST(I) .GT. ZSTOP(N))
            N=N+1
            IF (N .GT. NSTOP) GO TO 200
         END DO
         IF (ZLIST(I) .NE. ZSTOP(N)) THEN
            DO J=NSTOP,N,-1
               ZSTOP(J+1)=ZSTOP(J)
               SWLIST(J+1)=SWLIST(J)
            END DO
            SWLIST(N)=1
         END IF
  200    IF (NSTOP .EQ. 0) N=1
         ZSTOP(N)=ZLIST(I)
         NSTOP=NSTOP+1
         SWLIST(N)=SWLIST(N)*TYPE
      END DO
      RETURN
      END
C============================================================================
      SUBROUTINE EVENT(SWITCH,ICHNG,IRSTRT,IASTRT,IASTOP)
C     CHECKS SWLIST TO DETERMINE WHICH EVENTS OCCUR.
C     EVENT TYPE CODE: 2--PRINT RESTART DATA; 3--ATTENUATOR START;
C     5--ATTENUATOR STOP; 7--CIRCUIT CHANGE
C     -------------------------------------------------------------
      INTEGER SWITCH
      ICHNG=0
      IRSTRT=0
      IASTRT=0
      IASTOP=0
      TEST=MOD(SWITCH,2)
      IF (TEST .EQ. 0) IRSTRT=1
      TEST=MOD(SWITCH,3)
      IF (TEST .EQ. 0) IASTRT=1
      TEST=MOD(SWITCH,5)
      IF (TEST .EQ. 0) IASTOP=1
      TEST=MOD(SWITCH,7)
      IF (TEST .EQ. 0) ICHNG=1
      RETURN
      END
C============================================================================
$EMA //
      SUBROUTINE CHANGE(Y)
C     RECOMPUTES CIRCUIT PARAMETERS WHEN A CHANGE IS REACHED
C     -------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SCS,SWITCH,SWLIST,SETU
      DOUBLE PRECISION ICOS,IFUNCT,ISIN,I0,K1,K2,K3,I0GB,I1GB,K0GB,K1GB,
     &LA,LT,MAG,NORMZ
      COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),A0(3),B(3),BETA(3),
&BETA0(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
&DF(3),EBEAM,EBLNC,EM,EPSI,ESAT,ETA,EWAVE,EXP1,F(3),FILL,G(3),
&GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),I0,K1,K2,K3,
&LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
&PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
&RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
&THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,V0,W(3),WP,WPWC,
&YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
 NCHNG=NCHNG+1
 ANEW=CKTRAD(NCHNG)
 IF (ANEW .NE. SMALLA) THEN               !NEW HELIX RADIUS!
    FILL=FILL*SMALLA/ANEW                 !FILL FACTOR!
    V2=V0*(1.-I0*V0**(-1.5)*K1*(.5-LOG(FILL)))
    VREL=K2*(1.-1./(1.+K3*V2)**2)         !RELATIVISTIC VOLTAGE!
    UZERO=SQRT(VREL)*2.33502D+07          'BEAM VELOCITY (in/s)!
    DENOM=PI*EPSI*UZERO*BP*BP
    WP=SQRT(EM*I0*61024./DENOM)           !PLASMA FREQ. (rads/s)'
 END IF
 DO J=1,NUMF
    IF (PHVELN(NCHNG,1) .LT. 1) THEN      !PHASE VELOCITY!
       VPNEW=PHVELN(NCHNG,J)*SMALLC
       BETANU=VP(J)*BETA(J)/VPNEW
       GAMANU=SQRT(BETANU**2-BETA0(J)**2)
    ELSE                                  'TPI'
       GAMANU=GAMA(J)*ANEW*PHVELN(NCHNG,1)/(SMALLA*TPI)
       BETANU=SQRT(GAMANU**2+BETA0(J)**2)
       VPNEW=W(J)/BETANU
       TPI=PHVELN(NCHNG,1)
    END IF
    Z1=AXIMP(J)                           !OLD CKT IMPEDANCE'
    IF (AXIMPN(NCHNG,J) .EQ. 99) THEN     !TPI SCALING'
       AXIMP(J)=AXIMP(J)*(BETA(J)/BETANU)**3*(GAMANU/GAMA(J))**4*
&      (EXP(-.6664*GAMANU*ANEW)/EXP(-.6664*GAMA(J)*SMALLA))**3
    ELSE
       AXIMP(J)=AXIMPN(NCHNG,J)
    END IF
    Z2=AXIMP(J)                           'NEW CKT IMPEDANCE!
    SMALLA=ANEW                           !HELIX RADIUS (in)'
    GAMA(J)=GAMANU                        'RADIAL PROP. CONST.'
    BETA(J)=BETANU                        !PROPAGAITION CONST.'
    VP(J)=VPNEW                           'PHASE VELOCITY (in/s)'
    CALL BESSEL(I0GB,I1GB,K0GB,K1GB,GAMA(J)*BP)
    ZAVG=AXIMP(J)*(I0GB**2-I1GB**2        'AVERAGED IMPEDANCE'
    C(J)=(I0*ZAVG*.25/VREL)**(1./3.       'GAIN PARAMETER'
    B(J)=(UZERO/VP(J)-1.)/C(J)            'VELOCITY PARAMETER'
    RFLSS(J)=RFLSSN(NCHNG,J)              'CIRCUIT LOSS (dB/in)'
    LT=RFLSS(J)/LOG10(EXP(1.))
    D(J)=LT*.05/(W(J)/UZERO*C(J))         'LOSS PARAMETER'
    A(J)=A(J)*2*Z2/(Z1+Z2)
    G(J)=G(J)*2*Z2/(Z1+Z2)
    F(J)=F(J)*Z1/Z2+W(J)/(W(1)*C(1))*(1.-Z1/Z2)
 END DO
 BB=BP*BETA(1)
 AB=SMALLA*BETA(1)
 W1C1=W(1)*C(1)
 WPWC=WP/W1C1
 IF (SCS .GT. 0) THEN
    CALL SPACE(AB,BB,SCS,R,EXP1,CB,PI)
 ELSE
```

```
            WPWC=0
         END IF
         WQ=R*WP
         NORMZ=W1C1/UZERO
         YFINAL=ZF*NORMZ
         PRNT=DPC*NORMZ
         Y=Z*NORMZ
         RETURN
         END
C===========================================================================
$EMA //
      SUBROUTINE ATTEN(Y,FIRST,IASTOP)
C     ACCOMODATES FOR LOSS DUE TO AN ATTENUATOR, AND CALLS THE
C     DRIFT TUBE EQUATIONS FOR A SEVER
C     ---------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SCS,SWITCH,SWLIST,SETU
      DOUBLE PRECISION ICOS,IFUNCT,ISIN,I0,K1,K2,K3,LA,LMAX,LOG10E,
     &LOSS,MAG,NORMZ
      COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),A0(3),B(3),BETA(3),
     &BETA0(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
     &DF(3),EBEAM,EBLNC,EM,EPSI,ESAT,ETA,EWAVE,EXP1,F(3),FILL,G(3),
     &GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),I0,K1,K2,K3,
     &LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
     &PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
     &RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
     &THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,V0,W(3),WP,WPWC,
     &YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
      LMAX=170.
      LOG10E=LOG10(EXP(1.))
      NATTN=NATTN+1
      DO NSEC=1,20                      !20 ATTENUATOR SECTIONS!
         ND=ND+1
         YHALT=ZSTOP(ND)*NORMZ
         DO J=1,NUMF                    !RECOMPUTE LOSS PARAMETER!
            LOSS=RFLSS(J)+SQRT(W(J)/W(1))*LA(NSEC,NATTN)
            D(J)=LOSS*.05/(W(J)/UZERO*C(J)*LOG10E)   !LOSS PARAMETER!
         END DO
         IF (LA(NSEC,NATTN) .GT. LMAX) THEN   !TREAT AS A SEVER!
            DO I=1,NUMF
               A(I)=MAX(A0(I)*.01D0,A(I)*.01D0)
               THETA(I)=0
               G(I)=0
               F(I)=0
            END DO
            CALL DRIFT(Y)
            Z=Y/NORMZ
            CALL OUTPUT(FIRST)
            IF (LA(NSEC+1,NATTN) .LE. LMAX) THEN 'INITIAL CONDITIONS'
            DO I=1,NUMF
               J=I
               CALL FINT(PHI,U,C(1),J,TSIN,TCOS,NELECT,UTERM,PI)
               ISIN(I)=TSIN
               ICOS(I)=TCOS
            END DO
            W1C1=W(1)*C(1)
            DO N=1,NUMF              'SET INITIAL COND. FOR END OF SEVER'
               WNW1=W(N)/W(1)
               WNW2=WNW1*WNW1
               BTERM=(1.+C(N)*B(N))/C(1)
               DTERM=2.*C(N)*D(N)
               ZNORM=AXIMP(N)/AXIMP(1)
               THETA(N)=0.D0
               F(N)=WNW1/C(1)-WNW1*SQRT(ABS(BTERM*BTERM-ZNORM
     &         *BTERM/(A(N)*PI)*(ICOS(N)+DTERM*ISIN(N))))
               FTERM=F(N)-W(N)/W1C1
               G(N)=WNW2*BTERM/FTERM*(A(N)*C(N)*D(N)*BTERM+.5/PI
     &         *ZNORM*(ISIN(N)-DTERM*ICOS(N)))
            END DO
         END IF
         ELSE                      !TREAT AS AN ATTEN!
            CALL INTSUB(Y,FIRST)
         END IF
         SWITCH=SWLIST(ND)
         CALL EVENT(SWITCH,ICHNG,IRSTRT,IASTRT,IASTOP)
         IF (IRSTRT .EQ. 1) THEN
            CALL OUTRS(Y)
            ND=ND+1
         END IF
         IF (ICHNG .EQ. 1) THEN
            CALL CHANGE(Y)
            ND=ND+1
         END IF
      IF (IASTOP .EQ. 1) GO TO 10 !END OF ATTENUATOR!
      END DO
 10   DO J=1,NUMF                  !RESET LOSS PARAMETER!
         D(J)=RFLSS(J)*.05/(W(J)/UZERO*C(J)*LOG10E)
      END DO
      RETURN
      END
C=======================================================================
$EMA //
      SUBROUTINE OUTPUT(FIRST)
C     CONTROLS PRINTOUT AT EACH AXIAL POSITION
C     ----------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SCS,SWITCH,SWLIST,SETU
      DOUBLE PRECISION ICOS,IFUNCT,ISIN,I0,K1,K2,K3,LA,MAG,NORMZ,PDBI(3)
      COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),A0(3),B(3),BETA(3),
     &BETA0(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
     &DF(3),EBEAM,EBLNC,EM,EPSI,ESAT,ETA,EWAVE,EXP1,F(3),FILL,G(3),
     &GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),I0,K1,K2,K3,
     &LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
     &PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
     &RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
     &THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,V0,W(3),WP,WPWC,
     &YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
      LU=1
      PW=2.*C(1)*I0*VREL*A(1)*A(1)*(1.-C(1)*F(1))/(1+C(1)*B(1))
      PDB=10.*LOG10(ABS(PW))+30.
      DPDB=(PDB-OLDPDB)/(Z-OLDZ)
      OLDPDB=PDB
      OLDZ=Z
      ETA=PW/(I0*V0)
      PDB1=PDB
      PDBI(1)=PDB1
      IF ((DPDB .LE. 0) .AND. (PDB1 .GT. PSAT)) THEN   !NEW SAT. POWER!
         PSAT=PDB1
```

```
            GSAT=PDB1-PINI(1)
            ESAT=ETA*100.
            ZSAT=Z
         END IF
         DTDZ=F(1)/NORMZ
         IF ((NPR .EQ. 2) .OR. (NPR .EQ. 4))
     &        CALL ENERGY(A,A0,C,U,NUMF,NELECT,EWAVE,EBEAM,EBLNC)
         IF (FIRST .NE. 1) THEN
            WRITE(98,5)
            IF (NPR .EQ. 2 .OR. NPR .EQ. 4) WRITE(98,10)
            IF (NPR .EQ. 3 .OR. NPR .EQ. 4) WRITE(98,20)
    5       FORMAT(/5X,'Z',7X,'PDB',7X,'B',5X,'THETA',5X,'DPDB',6X,'DTDZ',
     &      6X,'ETA',6X,'D_')
   10       FORMAT(/1X'EBEAM',8X,'EWAVE',8X,'EBLNC_')
   20       FORMAT(/' PHI(I,1) ELECTRON PHASE ARRAY (FUNDAMENTAL)'/
     &      ' 2*C(1)*U(I) NORM. AC VELOCITY ARRAY')
            FIRST=1
         END IF
         WRITE(98,25)Z,PDB,B(1),THETA(1),DPDB,DTDZ,ETA,D(1)
   25    FORMAT(/1X,F7.3,F9.3,F8.2,F9.3,F10.3,F10.4,F8.4,F9.4)
         IF (NUMF .GT. 1) THEN
            DO J=2,NUMF
               PW=2.*C(1)*I0*VREL*A(J)*A(J)*(1.-C(J)*F(J))/(1+C(J)*B(J))
               PDB=10.*LOG10(ABS(PW))+30.
               PDBI(J)=PDB
               WRITE(98,30)PDB,B(J),THETA(J)
   30          FORMAT(8X,F9.3,F8.2,F9.3)
            END DO
         END IF
         WRITE(97,33)Z,(PDBI(N),N=1,NUMF),(THETA(M),M=1,NUMF)
         WRITE(LU,33)Z,(PDBI(N),N=1,NUMF)
         IF (NPR .EQ. 4) WRITE(96,32)Z,(PHI(I,1),I=1,NELECT)
   32    FORMAT(1X,F4.3,64(1X,F5.2))
   33    FORMAT(1X,F7.4,6(2X,F8.4))
         IF ((NPR .EQ. 2) .OR. (NPR .EQ. 4)) THEN
            WRITE(98,35)EBEAM,EWAVE,EBLNC
   35       FORMAT(1X,3(G12.5,2X))
         END IF
         IF ((NPR .EQ. 3) .OR. (NPR .EQ. 4)) THEN
            WRITE(98,45)(PHI(I,1),I=1,NELECT)
            WRITE(98,45)(2.*C(1)*U(I),I=1,NELECT)
   45       FORMAT(4(1X,G12.4,3X))
         END IF
         RETURN
         END
C=====================================================================
$EMA //
         SUBROUTINE OUTRS(Y)
C        PRINTS OUT RESTART DATA
C        -----------------------
         IMPLICIT DOUBLE PRECISION (A-H,O-Z)
         INTEGER SCS,SWITCH,SWLIST,SETU
         DOUBLE PRECISION ICOS,IFUNCT,ISIN,I0,I0GB,I1GB,K0GB,K1GB,K1,K2,K3,
     &LA,LT,MAG,NORMZ
         COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),A0(3),B(3),BETA(3),
     &BETA0(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
     &DF(3),EBEAM,EBLNC,EM,EPSI,ESAT,ETA,EWAVE,EXP1,F(3),FILL,G(3),
     &GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),I0,K1,K2,K3,
     &LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
     &PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
     &RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
     &THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,V0,W(3),WP,WPWC,
     &YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
      WRITE(98,10)Y
   10 FORMAT(//' RESTART INFORMATION FOR Y= ',F7.3/)
      WRITE(98,20)
   20 FORMAT(6X,'A',11X,'DA',8X,'THETA',7X,'DTHETA')
      DO I=1,NUMF
         WRITE(98,30)A(I),DA(I),THETA(I),DTHETA(I)
   30    FORMAT(4(1X,E10.4,1X))
      END DO
      WRITE(98,40)
   40 FORMAT(/' ELECTRON PHASE POSITIONS')
      DO I=1,NELECT
         WRITE(98,30)(PHI(I,J),J=1,NUMF)
      END DO
      WRITE(98,50)
   50 FORMAT(/' ELECTRON VELOCITIES')
      WRITE(98,30)(U(I),I=1,NELECT)
      IF (SCS .EQ. 2) THEN
         WRITE(98,60)(CB(I),I=1,5)
   60    FORMAT(/' COEFF. FOR POLYNOMIAL EXPANSION OF SPACE CHARGE'/1X,
     &         5(E10.4,2X))
      END IF
      RETURN
      END
C===========================================================================
      SUBROUTINE ENERGY(A,A0,C,U,NUMF,NELECT,EWAVE,EBEAM,EBLNC)
C     CHECKS FOR CONSERVATION OF ENERGY
C     --------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION A(NUMF),A0(NUMF),C(NUMF),U(NELECT)
      EMA A,A0,C,U,NUMF,NELECT,EWAVE,EBEAM,EBLNC
      EWAVE=0
      EBEAM=0
      DO I=1,NUMF
         EWAVE=A(I)*A(I)-A0(I)*A0(I)+EWAVE
      END DO
      EWAVE=2.*C(1)*EWAVE
      DO I=1,NELECT
         CTERM=1.+2.*C(1)*U(I)
         EBEAM=CTERM*CTERM+EBEAM
      END DO
      EBEAM=EBEAM/NELECT-1.
      EBLNC=ABS(EBEAM)-ABS(EWAVE)
      EBLNC=EBLNC/(ABS(EBEAM)+ABS(EWAVE))
      RETURN
      END
C========================================================================
      SUBROUTINE FINT(PHI,U,C1,J,TSIN,TCOS,NELECT,UTERM,PI)
C     CALCULATES THE BEAM CHARGE DENSITY
C     -----------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION PHI(64,3),U(NELECT),UTERM(NELECT)
      EMA PHI,U,C1,NELECT,UTERM,PI
      TWOPI=2.*PI
      TSIN=0
      TCOS=0
      DO I=1,NELECT
         UTERM(I)=1.+2.*C1*U(I)
         TSIN=TSIN+SIN(PHI(I,J))/UTERM(I)
         TCOS=TCOS+COS(PHI(I,J))/UTERM(I)
      END DO
      RN=FLOAT(NELECT)
      TSIN=TSIN*TWOPI/RN
      TCOS=TCOS*TWOPI/RN
      RETURN
      END
C=========================================================================
$EMA /TEMP/
      SUBROUTINE INT1(PHI,UTERM,EXP1,NELECT,IFUNCT,PI)
C     SOLVES EXACT SPACE CHARGE INTEGRAL FOR FORCE EQUATION
C     -----------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION IFUNCT(NELECT),IFUNSM,PHI(64,3),UTERM(NELECT)
      EMA PHI,UTERM,EXP1,NELECT,IFUNCT,PI
      COMMON /TEMP/ FTEMP(64,64)
      TWOPI=2*PI
      NM1=NELECT-1
      DO I=1,NM1
         IP1=I+1
         DO J=IP1,NELECT
            ZZ=PHI(I,1)-PHI(J,1)
            IF ((ZZ .GE. TWOPI) .OR. (ZZ .LT. 0))
     &         ZZ=ZZ-TWOPI*(INT(ZZ/TWOPI)-(1.-SIGN(1.D0,ZZ))*.5)
            IF ((ZZ .GE. TWOPI) .OR. (ZZ .LT. 0)) GO TO 500
            SINZZ=SIN(ZZ)
            COSZZ=COS(ZZ)
            TERM1=SINZZ/(EXP1-COSZZ)
            TERM2=SINZZ/(EXP1*EXP1-COSZZ)
            FTEMP(I,J)=((PI-ZZ)*.5-2.*ATAN(TERM1)+ATAN(TERM2))/TWOPI
            FTEMP(J,I)=-FTEMP(I,J)/UTERM(I)
            FTEMP(I,J)=FTEMP(I,J)/UTERM(J)
         END DO
      END DO
      DO I=1,NELECT
         IFUNSM=0
         DO J=1,NELECT
            IFUNSM=IFUNSM+FTEMP(I,J)
         END DO
         IFUNCT(I)=IFUNSM*TWOPI/FLOAT(NM1)
      END DO
 500  RETURN
      END
C=========================================================================
$EMA /TEMP/
      SUBROUTINE INT2(PHI,UTERM,CB,NELECT,IFUNCT,PI)
C     SOLVES THE POLYNOMIAL APPROXIMATION TO THE SPACE CHARGE
C     INTEGRAL FOR THE ELECTRON FORCE EQUATION
C     --------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION CB(5),IFUNCT(NELECT),IFUNSM,PHI(64,3),
     &UTERM(NELECT)
      EMA PHI,UTERM,CB,NELECT,IFUNCT,PI
      COMMON /TEMP/ FTEMP(64,64)
      TWOPI=2.*PI
      NM1=NELECT-1
      DO I=1,NM1
         IP1=I+1
         DO J=IP1,NELECT
            ZZ=PHI(I,1)-PHI(J,1)
            IF ((ZZ .GE. TWOPI) .OR. (ZZ .LT. 0))
     &         ZZ=ZZ-TWOPI*(INT(ZZ/TWOPI)-(1.-SIGN(1.D0,ZZ))*.5)
            IF ((ZZ .GE. TWOPI) .OR. (ZZ .LT. 0)) GO TO 500
            ZZ=ZZ/PI-1.
            FTEMP(I,J)=CB(3)*ZZ*ZZ*ZZ*ZZ*ZZ+CB(2)*ZZ*ZZ*ZZ+CB(1)*ZZ
            FTEMP(J,I)=-FTEMP(I,J)/UTERM(I)
            FTEMP(I,J)=FTEMP(I,J)/UTERM(J)
         END DO
      END DO
      DO I=1,NELECT
         IFUNSM=0
         DO J=1,NELECT
            IFUNSM=IFUNSM+FTEMP(I,J)
         END DO
         IFUNCT(I)=IFUNSM*TWOPI/FLOAT(NM1)
      END DO
  500 RETURN
      END
C===========================================================================
$EMA //
      SUBROUTINE DRFTEQ(Y,VALUES,EQNS)
C     PHASE AND FORCE EQUATIONS FOR THE DRIFT TUBE.  B(N) REPLACES
C     DTHETA IN THE PHASE EQUATION TO INSURE THE CORRECT INITIAL
C     PHASE FOR THE CIRCUIT WAVE IN THE NEXT SECTION
C     DPHI(I,N)--PHASE EQUATION; DU(I)--FORCE EQUATION
C     -----------------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SCS,SWITCH,SWLIST,SETU
      DOUBLE PRECISION ICOS,IFUNCT,ISIN,IO,K1,K2,K3,LA,MAG,NORMZ,
     &DU(64),DPHI(64,3),EQNS(268),VALUES(268)
      COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),A0(3),B(3),BETA(3),
     &BETA0(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
     &DF(3),EBEAM,EBLNC,EM,EPSI,ESAT,ETA,EWAVE,EXP1,F(3),FILL,G(3),
     &GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),IO,K1,K2,K3,
     &LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
     &PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
     &RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
     &THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,V0,W(3),WP,WPWC,
     &YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
      DO I=1,NELECT             !INITIAL VALUES FOR DRIFT EQNS!
         U(I)=VALUES(I)
      END DO
      DO J=1,NUMF
         DO I=1,NELECT
            PHI(I,J)=VALUES(I+J*NELECT)
         END DO
      END DO
      IF (SKPINT .EQ. 0) THEN  !COMPUTE SPACE CHARGE FIELDS & INTEGRAL!
         DO I=1,NUMF
            J=I
            CALL FINT(PHI,U,C(1),J,TSIN,TCOS,NELECT,UTERM,PI)
         END DO
         IF (SCS .EQ. 2) CALL INT2(PHI,UTERM,CB,NELECT,IFUNCT,PI)
         IF (SCS .EQ. 1) CALL INT1(PHI,UTERM,EXP1,NELECT,IFUNCT,PI)
      END IF
      SKPINT=SKPINT+1
      IF (SKPINT .GT. INTSKP) SKPINT=0
      DO I=1,NELECT            !DRIFT TUBE EQUATIONS!
         CTERM=1.+2.*C(1)*U(I)
         DO N=1,NUMF
            BTERM=1.+C(N)*B(N)
            DPHI(I,N)=B(N)+2.*W(N)*U(I)/(W(1)*CTERM)
            DU(I)=WPWC*WPWC*IFUNCT(I)/(BTERM*CTERM)
         END DO
      END DO
      DO I=1,NELECT            !STORE DRIFT EQUATIONS IN ARRAY EQNS!
         EQNS(I)=DU(I)
      END DO
      DO J=1,NUMF
         DO I=1,NELECT
            EQNS(I+J*NELECT)=DPHI(I,J)
         END DO
      END DO
      RETURN
      END
C=========================================================================
$EMA //
      SUBROUTINE DRIFT(Y)
C     SETS THE INITIAL CONDITIONS AND CALLS THE DIFFERENTIAL
C     EQUATION SOLVER FOR THE DRIFT TUBE EQUATIONS
C     ------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SCS,SWITCH,SWLIST,SETU
      DOUBLE PRECISION ICOS,IFUNCT,ISIN,IO,K1,K2,K3,LA,MAG,NORMZ,
     &VALUES(268)
      COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),A0(3),B(3),BETA(3),
     &BETA0(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
     &DF(3),EBEAM,EBLNC,EM,EPSI,ESAT,ETA,EWAVE,EXP1,F(3),FILL,G(3),
     &GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),IO,K1,K2,K3,
     &LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
     &PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
     &RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
     &THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,V0,W(3),WP,WPWC,
     &YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
      EXTERNAL DRFTEQ
      DO I=1,NELECT              !INITIAL VALUES FOR DRIFT EQNS!
         VALUES(I)=U(I)
      END DO
      DO J=1,NUMF
         DO I=1,NELECT
            VALUES(I+J*NELECT)=PHI(I,J)
         END DO
      END DO
      NEQN=NUMF*NELECT+NELECT
      ABSERR=1.D-09
      IFLAG=1
      RELERR=1.D-03
      CALL DESOLV(DRFTEQ,NEQN,VALUES,Y,YHALT,RELERR,ABSERR,IFLAG,SETU)
      DO I=1,NELECT
         U(I)=VALUES(I)
      END DO
      DO J=1,NUMF
         DO I=1,NELECT
            PHI(I,J)=VALUES(I+J*NELECT)
         END DO
      END DO
      RETURN
      END
C=======================================================================
$EMA //
      SUBROUTINE DIFFEQ(Y,VALUES,EQNS)
C     ROWE'S LARGE SIGNAL EQUATIONS
C     DA(N) & DG(N):       CIRCUIT WAVE AMPLITUDE EQUATION
C     DTHETA(N) & DF(N):  CIRCUIT WAVE PHASE EQUATION
C     DPHI(I,N):           PHASE RELATION EQUATION
C     DU(I):               ELECTRON FORCE EQUATION
C     -------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SCS,SWITCH,SWLIST,SETU
      DOUBLE PRECISION ICOS,IFUNCT,ISIN,I0,K1,K2,K3,LA,MAG,NORMZ,
     &DU(64),DPHI(64,3),DG(3),EQNS(268),VALUES(268)
      COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),A0(3),B(3),BETA(3),
     &BETA0(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
     &DF(3),EBEAM,EBLNC,EM,EPSI,ESAT,ETA,EWAVE,EXP1,F(3),FILL,G(3),
     &GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),I0,K1,K2,K3,
     &LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
     &PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
     &RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
     &THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,V0,W(3),WP,WPWC,
     &YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
      DO I=1,NELECT                 !INITIAL VALUES FOR DIFF. EQNS!
         U(I)=VALUES(I)
      END DO
      DO J=1,NUMF
         DO I=1,NELECT
            PHI(I,J)=VALUES(I+J*NELECT)
         END DO
      END DO
      K=NELECT*(NUMF+1)
      DO N=1,NUMF
         J=N+K
         G(N)=VALUES(J)
         F(N)=VALUES(J+NUMF)
         A(N)=VALUES(J+2*NUMF)
         THETA(N)=VALUES(J+3*NUMF)
      END DO
      IF (SKPINT .EQ. 0) THEN  !COMPUTE SPACE CHARGE FIELDS & INTEGRAL!
         DO I=1,NUMF
            J=I
            CALL FINT(PHI,U,C(1),J,TSIN,TCOS,NELECT,UTERM,PI)
            ISIN(I)=TSIN
            ICOS(I)=TCOS
         END DO
         IF (SCS .EQ. 2) CALL INT2(PHI,UTERM,CB,NELECT,IFUNCT,PI)
         IF (SCS .EQ. 1) CALL INT1(PHI,UTERM,EXP1,NELECT,IFUNCT,PI)
      END IF
      SKPINT=SKPINT+1
      IF (SKPINT .GT. INTSKP) SKPINT=0
      W1C1=W(1)*C(1)
      DO N=1,NUMF
         WNW1=W(N)/W(1)
         WNW2=WNW1*WNW1
         BTERM=(1.+C(N)*B(N))/C(1)
         DTERM=2.*C(N)*D(N)
         FTERM=F(N)-W(N)/W1C1
         ZNORM=AXIMP(N)/AXIMP(1)
         DA(N)=G(N)        !DIFFERENTIAL EQUATIONS!
         DTHETA(N)=F(N)
         DG(N)=A(N)*(FTERM**2-WNW2*BTERM**2)-ZNORM*WNW2*BTERM/PI*
     &         (ICOS(N)+DTERM*ISIN(N))
         DF(N)=DTERM*WNW2*BTERM**2-(2.*G(N)*FTERM+ZNORM*WNW2*BTERM/PI*
     &         (ISIN(N)-DTERM*ICOS(N)))/A(N)
         DO I=1,NELECT
            CTERM=1.+2.*C(1)*U(I)
            DPHI(I,N)=2.*WNW1*U(I)/CTERM-F(N)
         END DO
      END DO
      DO I=1,NELECT
         CTERM=1.+2.*C(1)*U(I)
         DU(I)=0
         DO N=1,NUMF
            FTERM=F(N)-W(N)/W1C1
            BTERM=1.+C(N)*B(N)
            CS=COS(PHI(I,N))
            SN=SIN(PHI(I,N))
            DU(I)=DU(I)+(G(N)*CS+A(N)*FTERM*SN)*C(1)
            DU(I)=(DU(I)+WPWC*WPWC*IFUNCT(I)/BTERM)/CTERM
         END DO
      END DO
      DO I=1,NELECT            !STORE DRIFT EQUATIONS IN ARRAY EQNS!
         EQNS(I)=DU(I)
      END DO
      DO J=1,NUMF
         DO I=1,NELECT
            EQNS(I+J*NELECT)=DPHI(I,J)
         END DO
      END DO
      K=NELECT*(NUMF+1)
      DO N=1,NUMF
         J=N+K
         EQNS(J)=DG(N)
         EQNS(J+NUMF)=DF(N)
         EQNS(J+2*NUMF)=DA(N)
         EQNS(J+3*NUMF)=DTHETA(N)
      END DO
      RETURN
      END
C========================================================================
$EMA //
      SUBROUTINE DIFF(Y)
C     SETS THE INITIAL CONDITIONS AND CALLS THE DIFFERENTIAL
C     EQUATION SOLVER FOR ROWE'S LARGE SIGNAL EQUATIONS
C     ------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      INTEGER SCS,SWITCH,SWLIST,SETU
      DOUBLE PRECISION ICOS,IFUNCT,ISIN,I0,K1,K2,K3,LA,MAG,NORMZ,
     &VALUES(268)
      COMMON A(3),ANG(3),AXIMP(3),AXIMPN(60,3),A0(3),B(3),BETA(3),
     &BETA0(3),BP,C(3),CB(5),CKTRAD(60),D(3),DA(3),DPC,DPDB,DTHETA(3),
     &DF(3),EBEAM,EBLNC,EM,EPSI,ESAT,ETA,EWAVE,EXP1,F(3),FILL,G(3),
     &GAMA(3),GSAT,ICOS(3),IFUNCT(64),INTSKP,ISIN(3),I0,K1,K2,K3,
     &LA(20,6),MAG(3),NATTN,NCHNG,ND,NELECT,NORMZ,NPR,NUMF,OLDPDB,OLDZ,
     &PDB,PDB1,PHI(64,3),PHVEL(3),PHVELN(60,3),PI,PINI(3),PRNT,PSAT,PW,
     &RFLSS(3),RFLSSN(60,3),SCS,SETU,SKPINT,SMALLA,SMALLC,SWLIST(140),
     &THETA(3),TPI,U(64),UTERM(64),UZERO,VP(3),VREL,V0,W(3),WP,WPWC,
     &YFINAL,YHALT,YPRINT,Z,ZF,ZSAT,ZSTOP(140)
      EXTERNAL DIFFEQ
      DO I=1,NELECT                  !INITIAL VALUES FOR DIFF EQNS!
        VALUES(I)=U(I)
      END DO
      DO J=1,NUMF
        DO I=1,NELECT
          VALUES(I+J*NELECT)=PHI(I,J)
        END DO
      END DO
      K=NELECT*(NUMF+1)
      DO N=1,NUMF
        J=N+K
        VALUES(J)=G(N)
        VALUES(J+NUMF)=F(N)
        VALUES(J+2*NUMF)=A(N)
        VALUES(J+3*NUMF)=THETA(N)
      END DO
      NEQN=4*NUMF+NELECT*NUMF+NELECT
      ABSERR=1.D-09
      IFLAG=1
      RELERR=1.D-03
      CALL DESOLV(DIFFEQ,NEQN,VALUES,Y,YPRINT,RELERR,ABSERR,IFLAG,SETU)
      DO I=1,NELECT
        U(I)=VALUES(I)
      END DO
      DO J=1,NUMF
        DO I=1,NELECT
          PHI(I,J)=VALUES(I+J*NELECT)
        END DO
      END DO
      K=NELECT*(NUMF+1)
      DO N=1,NUMF
        J=N+K
        G(N)=VALUES(J)
        F(N)=VALUES(J+NUMF)
        A(N)=VALUES(J+2*NUMF)
        THETA(N)=VALUES(J+3*NUMF)
      END DO
      RETURN
      END
C===============================================
$EMA /ABC/
      SUBROUTINE DESOLV(F,NEQN,Y,T,TOUT,RELERR,ABSERR,IFLAG,SETU)
C     INTEGRATES UP TO 268 DIFFERENTIAL EQUATIONS OVER THE INTERVAL
C     FROM T TO TOUT.  DESOLV CALLS THE INTEGRATOR STEP, AND THE
C     INTERPOLATION ROUTINE INTRP.  ALL THREE SUBROUTINES ARE
C     LISTED AND FULLY DOCUMENTED IN THE TEXT "COMPUTER SOLUTION
C     OF ORDINARY DIFFERENTIAL EQUATIONS: THE INITIAL VALUE PROBLEM"
C     BY L.F. SHAMPINE & M.K. GORDON.
C     ---------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      LOGICAL START,CRASH,STIFF
      INTEGER SETU
      DIMENSION Y(NEQN),PSI(12),YY(268),WT(268),PHI(268,16),
     &P(268),YP(268),YPOUT(268)
```

```
      COMMON /CDE/ YY,P,YP,PSI,U,FOURU
      COMMON /ABC/ PHI,WT,YPOUT
      EMA SETU,TOUT
      EXTERNAL F
      DATA MAXNUM/2000/
      EPS=MAX(RELERR,ABSERR)
      ISN=ISIGN(1,IFLAG)
      IFLAG=IABS(IFLAG)
      IF (SETU .EQ. 0) THEN
         CALL MACHIN(U)
         FOURU=4.*U
         SETU=1
      END IF
      IF (NEQN .LT. 1 .OR. NEQN .GT. 268) CALL BADIN
      IF (T .EQ. TOUT) CALL BADIN
      IF (RELERR .LT. 0.0 .OR. ABSERR .LT. 0.0) CALL BADIN
      IF (EPS .LE. 0 .OR. IFLAG .EQ. 0 ) CALL BADIN
      IF (IFLAG .NE. 1 .AND. T .NE. TOLD) CALL BADIN
      IF (IFLAG .LT. 1 .OR. IFLAG .GT. 5) CALL BADIN
      DEL=TOUT-T
      ABSDEL=ABS(DEL)
      TEND=T+10.0*DEL
      IF (ISN .LT. 0) TEND=TOUT
      NOSTEP=0
      KLE4=0
      STIFF=.FALSE.
      RELEPS=RELERR/EPS
      ABSEPS=ABSERR/EPS
      IF (IFLAG .EQ. 1 .OR. ISNOLD .LT. 0) THEN
         START=.TRUE.
         X=T
         DO L=1,NEQN
            YY(L)=Y(L)
         END DO
         DELSGN=SIGN(1.D0,DEL)
         H=SIGN(MAX(ABS(TOUT-X),FOURU*ABS(X)),TOUT-X)
      END IF
   10 IF (DELSGN*DEL .GT. 0.0 .AND. ABS(X-T) .GE. ABSDEL) THEN
         CALL INTRP(X,YY,TOUT,Y,YPOUT,NEQN,KOLD,PHI,PSI)
         IFLAG=2
         T=TOUT
         TOLD=T
         ISNOLD=ISN
         RETURN
      END IF
      IF (ISN .LE. 0 .OR. ABS(TOUT-X) .LT. FOURU*ABS(X)) THEN
         H=TOUT-X
         CALL F(X,YY,YP)
         DO L=1,NEQN
            Y(L)=YY(L)+H*YP(L)
         END DO
         IFLAG=2
         T=TOUT
         TOLD=T
         ISNOLD=ISN
         RETURN
      END IF
      IF (NOSTEP .GE. MAXNUM) THEN
         IFLAG=ISN*4
         IF (STIFF) IFLAG=ISN*5
         DO L=1,NEQN
            Y(L)=YY(L)
         END DO
         T=X
         TOLD=T
         ISNOLD=1
         RETURN
      END IF
      H=SIGN(MIN(ABS(H),ABS(TEND-X)),H)
      DO L=1,NEQN
         WT(L)=RELEPS*ABS(YY(L))+ABSEPS
      END DO
      CALL STEP(X,YY,F,NEQN,H,EPS,WT,START,HOLD,K,KOLD,CRASH,PHI,P,
     &          YP,PSI,U)
      IF (CRASH) THEN
         IFLAG=ISN*3
         RELERR=EPS*RELEPS
         ABSERR=EPS*ABSEPS
         DO L=1,NEQN
            Y(L)=YY(L)
         END DO
         T=X
         TOLD=T
         ISNOLD=1
         RETURN
      END IF
      NOSTEP=NOSTEP+1
      KLE4=KLE4+1
      IF (KOLD .GT. 4) KLE4=0
      IF (KLE4 .GE. 50) STIFF=.TRUE.
      GO TO 10
      END
C===========================================================================
      SUBROUTINE BADIN
C     WARNS USER OF BAD INPUT TO DESOLV AND STOPS PROGRAM EXECUTION
C     ---------------------------------------------------------------
      LU=1
      WRITE(LU,10)
   10 FORMAT(' INPUT TO THE DIFFERENTIAL EQUATION SOLVER IS BAD!')
      STOP
      END
C===========================================================================
      SUBROUTINE MACHIN(U)
C     CALCULATES THE SMALLEST POSITIVE NUMBER U, SUCH THAT
C     1.0+U > 1.0.  U IS COMPUTED APPROXIMATELY AS A POWER
C     OF 1/2
C     -----------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      HALFU=.5
   50 TEMP1=1.+HALFU
      IF (TEMP1 .LE. 1.) GO TO 100
      HALFU=.5*HALFU
      GO TO 50
  100 U=2.*HALFU
      RETURN
      END
C===========================================================================
      SUBROUTINE STEP(X,Y,F,NEQN,H,EPS,WT,START,HOLD,K,KOLD,CRASH,
     &                PHI,P,YP,PSI,U)
C     INTEGRATES THE DIFFERENTIAL EQUATIONS OVER A STEP (X TO X+H) USING
C     THE MODIFIED DIVIDED DIFFERENCE FORM OF THE ADAMS PECE FORMLAS.
C     ------------------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      LOGICAL START,CRASH,PHASE1,NORND
      DIMENSION Y(NEQN),WT(NEQN),PHI(NEQN,16),P(NEQN),YP(NEQN),
     &PSI(12),GSTR(13),TWO(13),ALPHA(12),BETA(12),SIG(13),W(12),
     &V(12),G(13)
      COMMON /CSTEP/ GSTR,TWO,ALPHA,BETA,SIG,W,V,G
      EMA PHI,WT
      EXTERNAL F
      TWO(1)=2.
      TWO(2)=4.
      TWO(3)=8.
      TWO(4)=16.
      TWO(5)=32.
      TWO(6)=64.
      TWO(7)=128.
      TWO(8)=256.
      TWO(9)=512.
      TWO(10)=1024.
      TWO(11)=2048.
      TWO(12)=4096.
      TWO(13)=8192.
      GSTR(1)=.5
      GSTR(2)=.0833
      GSTR(3)=.0417
      GSTR(4)=.0264
      GSTR(5)=.0188
      GSTR(6)=.0143
      GSTR(7)=.0114
      GSTR(8)=.00936
      GSTR(9)=.00789
      GSTR(10)=.00679
      GSTR(11)=.00592
      GSTR(12)=.00524
      GSTR(13)=.00468
      G(1)=1.
      G(2)=.5
      SIG(1)=1.
      CRASH=.TRUE.
      TWOU=2.*U
      FOURU=4.*U
      IF (ABS(H) .LT. FOURU*ABS(X)) THEN
         H=SIGN(FOURU*ABS(X),H)
         RETURN
      END IF
      P5EPS=.5*EPS
      ROUND=0.
      DO L=1,NEQN
         ROUND=ROUND+(Y(L)/WT(L))**2
      END DO
      ROUND=TWOU*SQRT(ROUND)
      IF (P5EPS .LT. ROUND) THEN
         EPS=2.*ROUND*(1.+FOURU)
         RETURN
      END IF
      CRASH=.FALSE.
```

```
       IF (.NOT. START)  GO TO 10
       CALL F(X,Y,YP)
       SUM=0.
       DO L=1,NEQN
          PHI(L,1)=YP(L)
          PHI(L,2)=0.
          SUM=SUM+(YP(L)/WT(L))**2
       END DO
       SUM=SQRT(SUM)
       ABSH=ABS(H)
       IF (EPS .LT. 16.*SUM*H*H)  ABSH=.25*SQRT(EPS/SUM)
       H=SIGN(MAX(ABSH,FOURU*ABS(X)),H)
       HOLD=0.
       K=1
       KOLD=0
       START=.FALSE.
       PHASE1=.TRUE.
       NORND=.TRUE.
       IF (P5EPS .GT. 100.*ROUND)  GO TO 10
       NORND=.FALSE.
       DO L=1,NEQN
          PHI(L,15)=0.
       END DO
10     IFAIL=0
20     KP1=K+1
       KP2=K+2
       KM1=K-1
       KM2=K-2
       IF (H .NE. HOLD)  NS=0
       NS=MIN0(NS+1,KOLD+1)
       NSP1=NS+1
       IF (K .LT. NS)  GO TO 50
       BETA(NS)=1.
       REALNS=NS
       ALPHA(NS)=1./REALNS
       TEMP1=H*REALNS
       SIG(NSP1)=1.
       IF (K .GE. NSP1) THEN
          DO I=NSP1,K
             IM1=I-1
             TEMP2=PSI(IM1)
             PSI(IM1)=TEMP1
             BETA(I)=BETA(IM1)*PSI(IM1)/TEMP2
             TEMP1=TEMP2+H
             ALPHA(I)=H/TEMP1
             REALI=I
             SIG(I+1)=REALI*ALPHA(I)*SIG(I)
          END DO
       END IF
       PSI(K)=TEMP1
       IF (NS .LE. 1) THEN
          DO IQ=1,K
             TEMP3=IQ*(IQ+1)
             V(IQ)=1./TEMP3
             W(IQ)=V(IQ)
          END DO
          GO TO 40
       END IF
       IF (K .LE. KOLD)  GO TO 30
```

```
      TEMP4=K*KP1
      V(K)=1./TEMP4
      NSM2=NS-2
      IF (NSM2 .LT. 1) GO TO 30
      DO J=1,NSM2
         I=K-J
         V(I)=V(I)-ALPHA(J+1)*V(I+1)
      END DO
30    LIMIT1=KP1-NS
      TEMP5=ALPHA(NS)
      DO IQ=1,LIMIT1
         V(IQ)=V(IQ)-TEMP5*V(IQ+1)
         W(IQ)=V(IQ)
      END DO
      G(NSP1)=W(1)
40    NSP2=NS+2
      IF (KP1 .GE. NSP2) THEN
         DO I=NSP2,KP1
            LIMIT2=KP2-I
            TEMP6=ALPHA(I-1)
            DO IQ=1,LIMIT2
               W(IQ)=W(IQ)-TEMP6*W(IQ+1)
            END DO
            G(I)=W(1)
         END DO
      END IF
50    IF (K .GE. NSP1) THEN
         DO I=NSP1,K
            TEMP1=BETA(I)
            DO L=1,NEQN
               PHI(L,I)=TEMP1*PHI(L,I)
            END DO
         END DO
      END IF
      DO L=1,NEQN
         PHI(L,KP2)=PHI(L,KP1)
         PHI(L,KP1)=0.
         P(L)=0.
      END DO
      DO J=1,K
         I=KP1-J
         IP1=I+1
         TEMP2=G(I)
         DO L=1,NEQN
            P(L)=P(L)+TEMP2*PHI(L,I)
            PHI(L,I)=PHI(L,I)+PHI(L,IP1)
         END DO
      END DO
      IF (.NOT. NORND) THEN
         DO L=1,NEQN
            TAU=H*P(L)-PHI(L,15)
            P(L)=Y(L)+TAU
            PHI(L,16)=(P(L)-Y(L))-TAU
         END DO
         GO TO 60
      END IF
      DO L=1,NEQN
         P(L)=Y(L)+H*P(L)
      END DO
```

```
 60     XOLD=X
        X=X+H
        ABSH=ABS(H)
        CALL F(X,P,YP)
        ERKM2=0.
        ERKM1=0.
        ERK=0.
        DO L=1,NEQN
           TEMP3=1./WT(L)
           TEMP4=YP(L)-PHI(L,1)
           IF (KM2) 90,80,70
70        ERKM2=ERKM2+((PHI(L,KM1)+TEMP4)*TEMP3)**2
80        ERKM1=ERKM1+((PHI(L,K)+TEMP4)*TEMP3)**2
90        ERK=ERK+(TEMP4*TEMP3)**2
        END DO
        IF (KM2) 120,110,100
100     ERKM2=ABSH*SIG(KM1)*GSTR(KM2)*SQRT(ERKM2)
110     ERKM1=ABSH*SIG(K)*GSTR(KM1)*SQRT(ERKM1)
120     TEMP5=ABSH*SQRT(ERK)
        ERR=TEMP5*(G(K)-G(KP1))
        ERK=TEMP5*SIG(KP1)*GSTR(K)
        KNEW=K
        IF (KM2) 150,140,130
130     IF (MAX(ERKM1,ERKM2) .LE. ERK) KNEW=KM1
        GO TO 150
140     IF (ERKM1 .LE. .5*ERK) KNEW=KM1
150     IF (ERR .LE. EPS) GO TO 190
        PHASE1=.FALSE.
        X=XOLD
        DO I=1,K
           TEMP1=1./BETA(I)
           IP1=I+1
           DO L=1,NEQN
              PHI(L,I)=TEMP1*(PHI(L,I)-PHI(L,IP1))
           END DO
        END DO
        IF (K .GE. 2) THEN
           DO I=2,K
              PSI(I-1)=PSI(I)-H
           END DO
        END IF
        IFAIL=IFAIL+1
        TEMP2=.5
        IF (IFAIL-3) 180,170,160
 160    IF (P5EPS .LT. .25*ERK) TEMP2=SQRT(P5EPS/ERK)
 170    KNEW=1
 180    H=TEMP2*H
        K=KNEW
        IF (ABS(H) .LT. FOURU*ABS(X)) THEN
           CRASH=.TRUE.
           H=SIGN(FOURU*ABS(X),H)
           EPS=EPS+EPS
           RETURN
        END IF
        GO TO 20
 190    KOLD=K
        HOLD=H
        TEMP1=H*G(KP1)
        IF (NORND) GO TO 200
```

```
      DO L=1,NEQN
         RHO=TEMP1*(YP(L)-PHI(L,1))-PHI(L,16)
         Y(L)=P(L)+RHO
         PHI(L,15)=(Y(L)-P(L))-RHO
      END DO
      GO TO 210
200   DO L=1,NEQN
         Y(L)=P(L)+TEMP1*(YP(L)-PHI(L,1))
      END DO
210   CALL F(X,Y,YP)
      DO L=1,NEQN
         PHI(L,KP1)=YP(L)-PHI(L,1)
         PHI(L,KP2)=PHI(L,KP1)-PHI(L,KP2)
      END DO
      DO I=1,K
         DO L=1,NEQN
            PHI(L,I)=PHI(L,I)+PHI(L,KP1)
         END DO
      END DO
      ERKP1=0.
      IF (KNEW .EQ. KM1 .OR. K .EQ. 12) PHASE1=.FALSE.
      IF (PHASE1) GO TO 220
      IF (KNEW .EQ. KM1) GO TO 230
      IF (KP1 .GT. NS) GO TO 240
      DO L=1,NEQN
         ERKP1=ERKP1+(PHI(L,KP2)/WT(L))**2
      END DO
      ERKP1=ABSH*GSTR(KP1)*SQRT(ERKP1)
      IF (K .LE. 1) THEN
         IF (ERKP1 .GE. .5*ERK) GO TO 240
         GO TO 220
      END IF
      IF (ERKM1 .LE. MIN(ERK,ERKP1)) GO TO 230
      IF (ERKP1 .GE. ERK .OR. K .EQ. 12) GO TO 240
220   K=KP1
      ERK=ERKP1
      GO TO 240
230   K=KM1
      ERK=ERKM1
240   HNEW=H+H
      IF (PHASE1) GO TO 250
      IF (P5EPS .GE. ERK*TWO(K+1)) GO TO 250
      HNEW=H
      IF (P5EPS .GE. ERK) GO TO 250
      TEMP2=K+1
      R=(P5EPS/ERK)**(1./TEMP2)
      HNEW=ABSH*MAX(.5D0,MIN(.9D0,R))
      HNEW=SIGN(MAX(HNEW,FOURU*ABS(X)),H)
250   H=HNEW
      RETURN
      END
C ----------------------------------------------------------------------
      SUBROUTINE INTRP(X,Y,XOUT,YOUT,YPOUT,NEQN,KOLD,PHI,PSI)
C     SUBROUTINE STEP APPROXIMATES THE SOLUTION NEAR X BY A
C     POLYNOMIAL.  SUBROUTINE INTRP APPROXIMATES THE SOLUTION
C     AT XOUT BY EVALUATNG THE POLYNOMIAL THERE.
C     ------------------------------------------------------------------
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Y(NEQN),YOUT(NEQN),YPOUT(NEQN),PHI(NEQN,16),
&PSI(12),G(13),W(13),RHO(13)
 EMA PHI,XOUT,YPOUT
 G(1)=1.
 RHO(1)=1.
 HI=XOUT-X
 KI=KOLD+1
 KIP1=KI+1
 DO I=1,KI
    TEMP1=I
    W(I)=1./TEMP1
 END DO
 TERM=0.
 DO J=2,KI
    JM1=J-1
    PSIJM1=PSI(JM1)
    GAMMA=(HI+TERM)/PSIJM1
    ETA=HI/PSIJM1
    LIMIT1=KIP1-J
    DO I=1,LIMIT1
       W(I)=GAMMA*W(I)-ETA*W(I+1)
    END DO
    G(J)=W(1)
    RHO(J)=GAMMA*RHO(JM1)
    TERM=PSIJM1
 END DO
 DO L=1,NEQN
    YPOUT(L)=0.
    YOUT(L)=0.
 END DO
 DO J=1,KI
    I=KIP1-J
    TEMP2=G(I)
    TEMP3=RHO(I)
    DO L=1,NEQN
       YOUT(L)=YOUT(L)+TEMP2*PHI(L,I)
       YPOUT(L)=YPOUT(L)+TEMP3*PHI(L,I)
    END DO
 END DO
 DO L=1,NEQN
    YOUT(L)=Y(L)+HI*YOUT(L)
 END DO
 RETURN
 END
```

APPENDIX B

TWT1D USER'S MANUAL

1. Introduction

TWT1D is a one-dimensional large-signal analysis program for helix traveling-wave tubes based on the theory developed by J. E. Rowe and M. K. Scherba. The program can be used to predict the performance of a helix traveling-wave tube before it is assembled. The input to the program consists of information about the beam, the input signals, and the helix. Parameters describing the helix are typically obtained from cold circuit testing, while information about the beam may come from beam analyzer results or computer design data. Prior to running TWT1D, the engineer creates an input data file which lists these parameters. This manual describes how this file is to be created and how to run TWT1D. Complete documentation for the program, including a source code listing, is given in AFTER report No. 17, available from the Electrical Engineering Department at the University of Utah.

2. Input Data File

The input file should be organized along the lines of Fig. A.1. The number of lines in the file is variable and is determined by the number of frequencies, velocity steps, attenuators, etc., present in the tube being modeled. The key following Fig. A.1 describes each of the variables listed in the input file, and Fig. A.2 shows a sample input file. Since the input files can become lengthy, it is strongly recommended that before using TWT1D, the input data file be checked with the program TWTIN. TWTIN is a short program which reads TWTID input files and lists the parameters so that the engineer can verify the input being used. The sample input file shown in Fig. A.3 was checked with TWTIN, and the result is shown in Fig. A.4.

```
Line         Data

  1          VO IO RAD FILL TPI ZF
  2          NUMF NUMC NUMA NUMR
  3          FI(1) PINI(1) PHVEL(1) AXIMP(1) RFLSS(1) THETA(1)
  4          FI(2) PINI(2)    .        .        .        .
  5          FI(3) PINI(3)    .        .        .        .
  6          ZC(1) CKTRAD(1) TAPER(1)
  7          ZC(2) CKTRAD(2) TAPER(2)
  .             .       .        .
  .             .       .        .
 11          ZC(6) CKTRAD(6) TAPER(6)
 12          PHVELN(1,1) PHVELN(1,2) PHVELN(1,3).....PHVELN(6,3)
 13          AXIMPN(1,1) AXIMPN(1,2) AXIMPN(1,3).....AXIMPN(6,3)
 14          RFLSSN(1,1) RFLSSN(1,2) RFLSSN(1,3).....RFLSSN(6,3)
 15          ZATNS(1) ZATNE(1) LAMAX(1) ATYPE(1)
 16          ZATNS(2) ZATNE(2) LAMAX(2) ATYPE(2)
  .              .        .        .        .
  .              .        .        .        .
 20          ZATNS(6) ZATNE(6) LAMAX(6) ATYPE(6)
 21          ZRST(1) ZRST(2) ZRST(3)............ZRST(10)
 22          SCS NELECT DPC NPR INTSKP RSTART FDUMMY
```

Fig. A.1. Input data file organization.

KEY TO INPUT DATA

Line	Variable	Description
1	VO	Dc beam voltage (volts)
	IO	Dc beam current (amps)
	RAD	Mean helix radius (in)
	FILL	Beam fill factor
	TPI	Helix pitch (turns/in)
	ZF	Length of circuit (in)
2	NUMF	Number of input frequencies (3 maximum)
	NUMC	Number of circuit changes and velocity tapers (6 maximum)
	NUMA	Number of attenuators (6 maximum)
	NUMR	Number of restart data printouts (10 maximum). The restart printout contains all the information that is needed to restart a run.
3-5	FI(i)	Frequency of ith signal (GHz)
	PINI(i)	Input power of ith signal (dBm)
	PHVEL(i)	Normalized phase velocity of ith signal $\left(V_p/c\right)$
	AXIMP(i)	Axial coupling impedance of ith signal (ohms)
	RFLSS(i)	RF circuit loss for ith signal (dB/in)
	THETA(i)	Injection angle of ith signal (degrees)
6-11	ZC(i)	Position of ith velocity step or taper (in)
	CKTRAD(i)	Radius of circuit beyond ith change (in)
	TAPER(i)	Length of velocity taper (in). Use 0 for velocity step.
12	PHVELN(i,j)	Normalized phase velocity of TPI of the helix for jth signal beyond ith change. If the TPI is used, the characteristics of the new section are calculated by scaling the phase

velocity and impedance values from the previous section. When the TPI is specified, use an axial impedance of 99. TPI scaling can only be used when the TPI of the previous section is given. Also, TPI scaling will not work when going from a vaned to an unvaned circuit, or vice versa.

13 AXIMPN(i,j) Axial impedance for jth signal beyond ith change (ohms)

14 RFLSSN(i,j) RF circuit loss for jth signal beyond ith change (dB/in)

15-20 ZATNS(i) Start position of ith attenuator (in)

ZATNE(i) Stop position of ith attenuator (in)

LAMAX(i) Maximum loss of ith attenuator (dB/in)

ATYPE(i) Type of ith attenuator (1, 2, 3, or 4). See Fig. A.2 for the different attenuator shapes available.

21 ZRST(i) Position of ith restart data printout (in)

22 SCS Space-charge switch:

0 = no space charge
1 = full space charge
2 = polynomial approximation to space charge

The space-charge switch determines how the repulsive force between electrons is calculated. If the tube has low beam current density, or quantitative results are not required, then it is appropriate to ignore the space charge and set SCS to 0. If the tube has few circuit changes and a small step size, the polynomial approximation to the space charge will give reasonable results, and SCS should be set to 2. For highest accuracy, SCS should be set to 1 so Rowe's full space-charge expression is used.

NELECT Number of electron disks (16, 32, or 64). Use 16 for one input signal; 32 for two input signals, and 64 for three input signals.

DPC Printout interval (in)

NPR

Printout option 1, 2, 3, or 4 (see Section IV for description)

INTSKP

Number of skips between evaluation of space-charge function. Since the electron positions do not vary rapidly, the space-charge function can be evaluated intermittently to reduce computation time.

RSTART

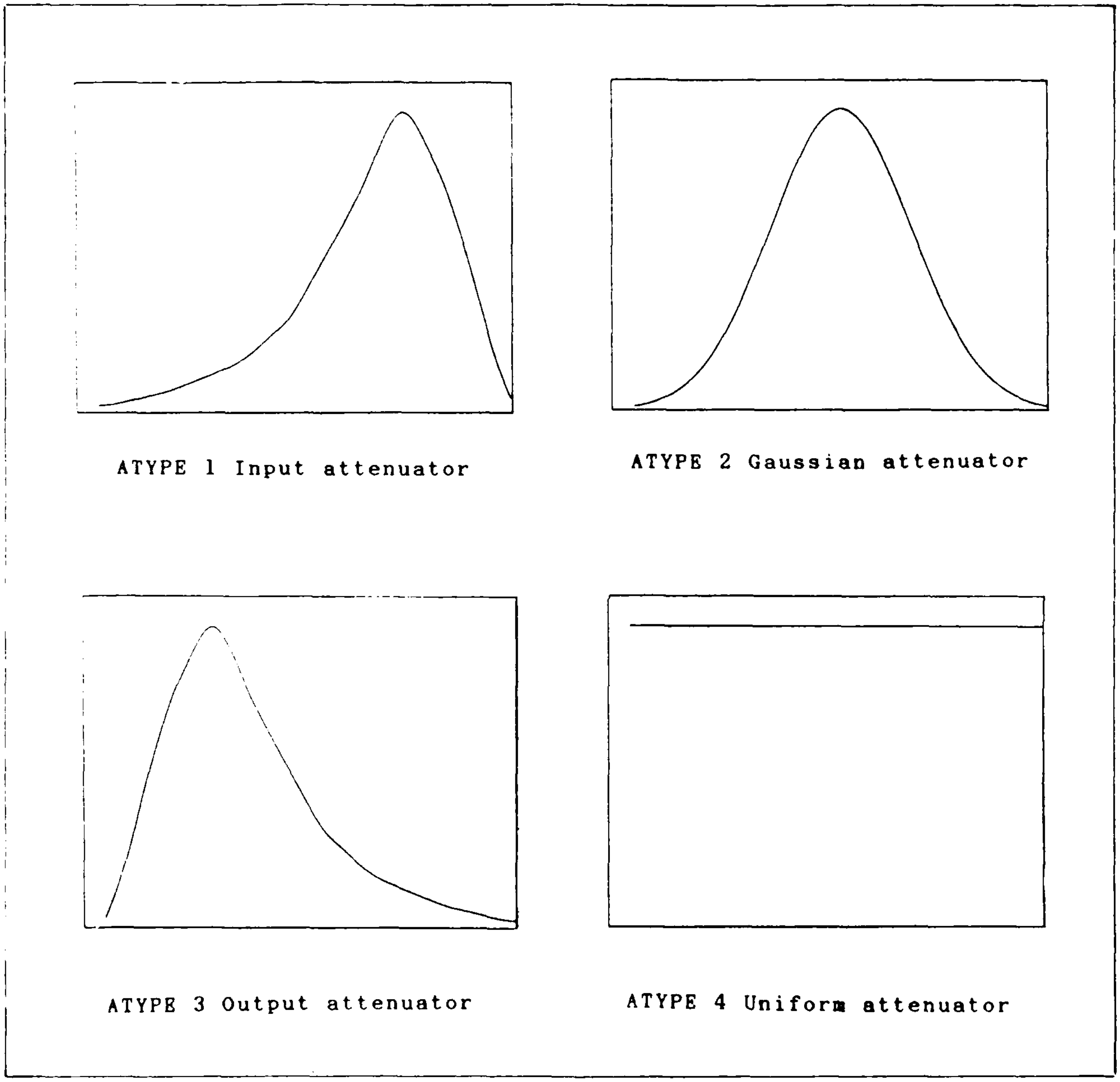
Restart data input switch:

0 = no restart data
1 = read restart data

Set to 1 if the run is to be restarted. A second data set must be included after the first one.

FDUMMY

Dummy frequency (GHz). Any data set, which specifies multiple frequencies which are not harmonically related, must also specify a dummy frequency. This dummy frequency is the highest frequency of which all input signals are harmonics. The lower this frequency, the less accurate the results will be. The dummy frequency is not followed, as it is an artificial construction.



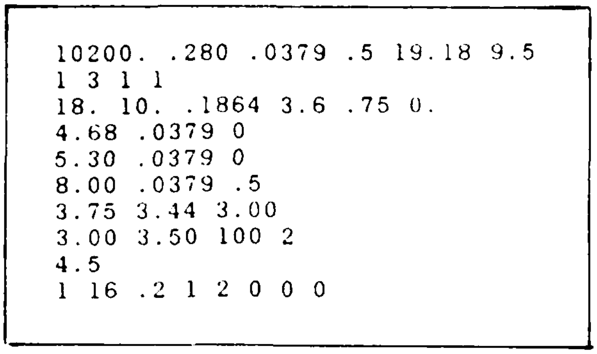
ATYPE 1 Input attenuator

ATYPE 2 Gaussian attenuator

ATYPE 3 Output attenuator

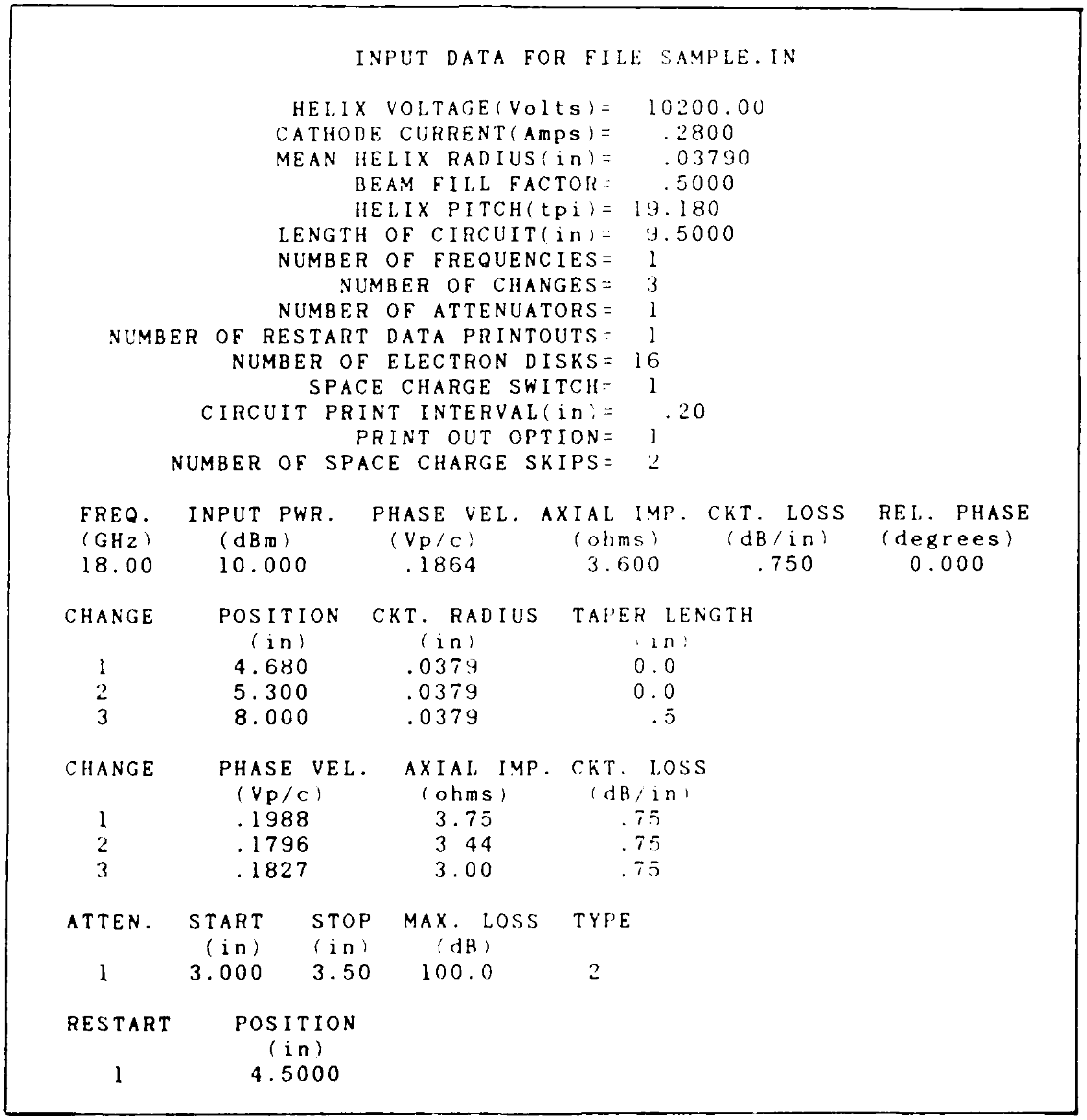
ATYPE 4 Uniform attenuator

Fig. A.2. Attenuator shapes.



```
10200. .280 .0379 .5 19.18 9.5
1 3 1 1
18. 10. .1864 3.6 .75 0.
4.68 .0379 0
5.30 .0379 0
8.00 .0379 .5
3.75 3.44 3.00
3.00 3.50 100 2
4.5
1 16 .2 1 2 0 0 0
```

Fig. A.3. Sample input file.

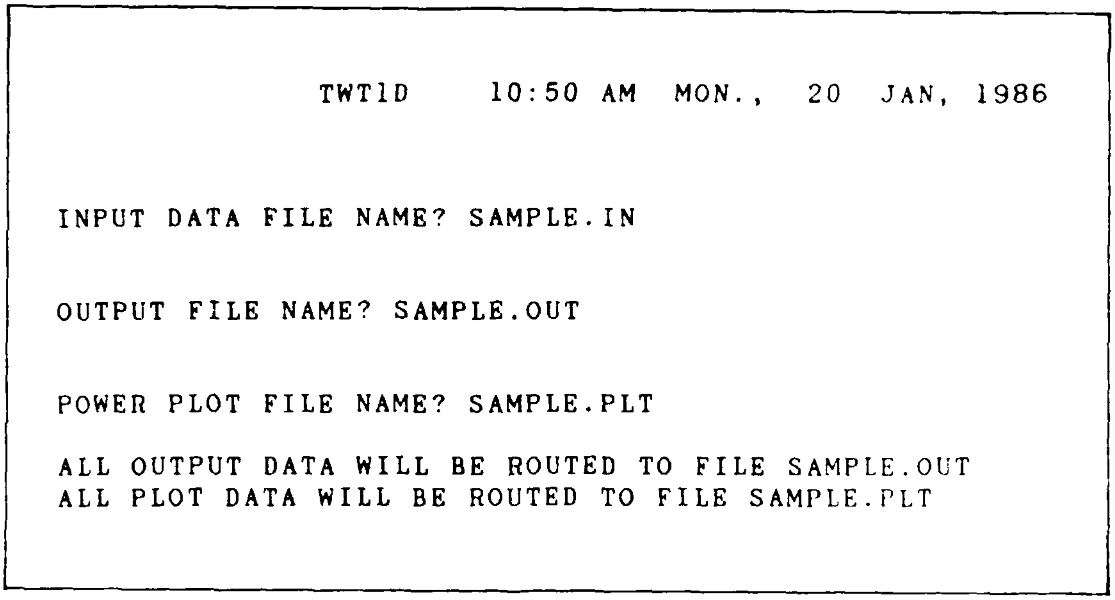


```
                    INPUT DATA FOR FILE SAMPLE.IN

                 HELIX VOLTAGE(Volts)=  10200.00
                CATHODE CURRENT(Amps)=    .2800
                MEAN HELIX RADIUS(in)=    .03790
                     BEAM FILL FACTOR=    .5000
                     HELIX PITCH(tpi)= 19.180
                LENGTH OF CIRCUIT(in)=  9.5000
                NUMBER OF FREQUENCIES=  1
                    NUMBER OF CHANGES=  3
                NUMBER OF ATTENUATORS=  1
     NUMBER OF RESTART DATA PRINTOUTS=  1
             NUMBER OF ELECTRON DISKS= 16
                  SPACE CHARGE SWITCH=  1
           CIRCUIT PRINT INTERVAL(in)=   .20
                     PRINT OUT OPTION=  1
         NUMBER OF SPACE CHARGE SKIPS=  2

 FREQ.   INPUT PWR.   PHASE VEL.  AXIAL IMP.  CKT. LOSS   REL. PHASE
 (GHz)     (dBm)        (Vp/c)      (ohms)     (dB/in)    (degrees)
 18.00    10.000        .1864       3.600       .750        0.000

CHANGE     POSITION   CKT. RADIUS   TAPER LENGTH
             (in)        (in)           (in)
  1          4.680       .0379           0.0
  2          5.300       .0379           0.0
  3          8.000       .0379            .5

CHANGE     PHASE VEL.   AXIAL IMP.  CKT. LOSS
             (Vp/c)      (ohms)      (dB/in)
  1          .1988        3.75         .75
  2          .1796        3.44         .75
  3          .1827        3.00         .75

ATTEN.   START    STOP   MAX. LOSS   TYPE
          (in)    (in)     (dB)
  1      3.000    3.50    100.0        2

RESTART     POSITION
              (in)
   1         4.5000
```

Fig. A.4. TWTIN run for sample input file.

3. Running the Program

Once an input file has been created and checked, the engineer is ready to run TWT1D. When run, the program prompts the user for the name of the input data file, the output data file, and the plot file. The output data file is where all the data from the run will be stored, and the plot file will contain the data needed to make plots of power versus position, and phase angle versus position. Although any name can be used for the output and plot files, the user is cautioned against using names of existing files. That can result in the original contents of those files being lost. It is recommended that the names of the output and plot files be related to each other so that it is easy to recall which plot file corresponds to which output file. For example, if the input file is called RUN1.DAT, the output and plot files could be named RUN1,OUT and RUN1.PLT, respectively. Figure A.5 shows the terminal display for a TWT1D run. As the program runs, the axial position and power will be printed to the terminal.

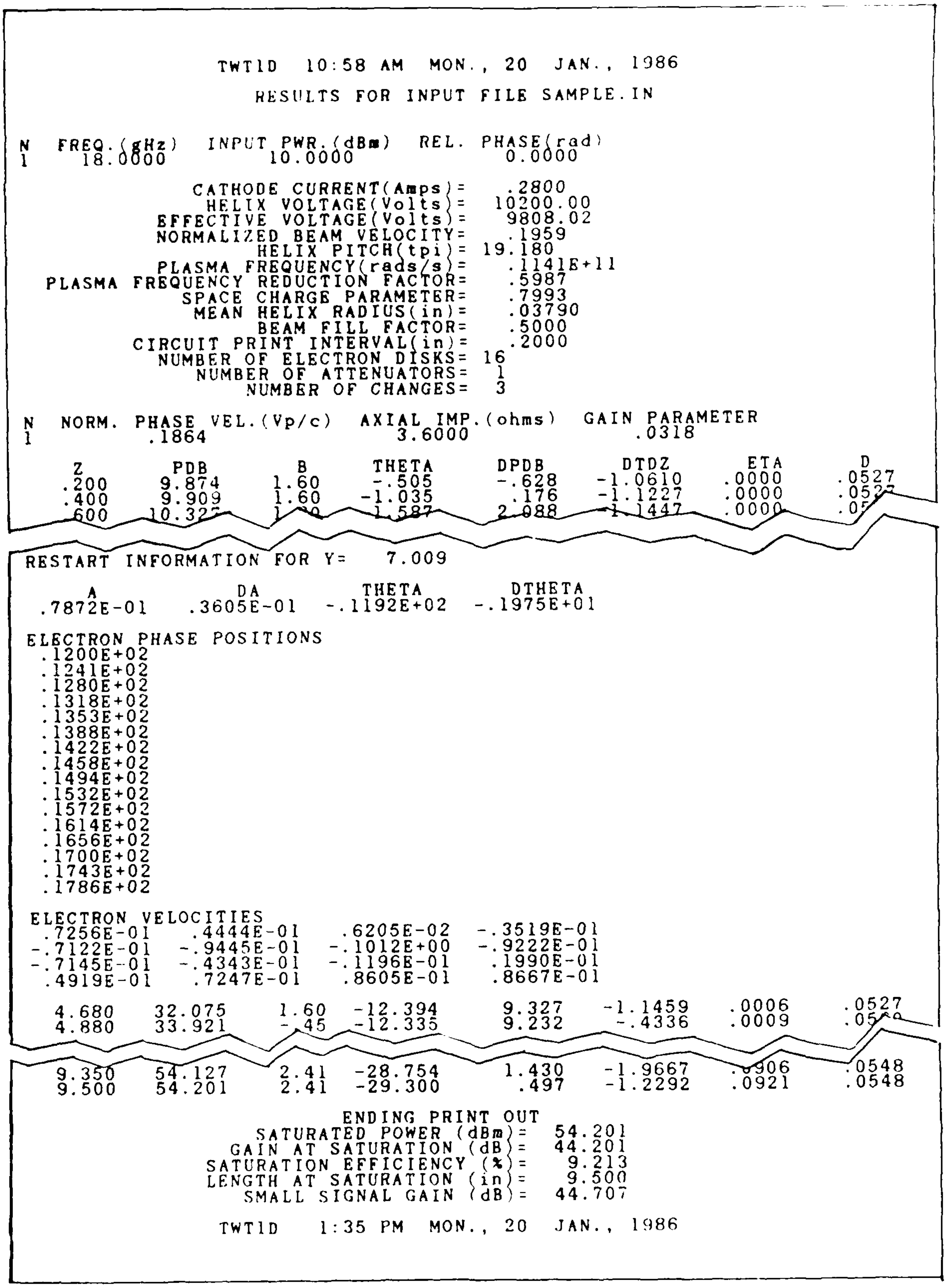


```
                     TWT1D     10:50 AM  MON.,   20  JAN, 1986


INPUT DATA FILE NAME? SAMPLE.IN


OUTPUT FILE NAME? SAMPLE.OUT


POWER PLOT FILE NAME? SAMPLE.PLT

ALL OUTPUT DATA WILL BE ROUTED TO FILE SAMPLE.OUT
ALL PLOT DATA WILL BE ROUTED TO FILE SAMPLE.PLT
```

Fig. A.5. Terminal display for a TWT1D run.
(User's input is underlined.)

4. Output Data File

The TWT1D output file for the sample input file is shown in Fig. A.6. Notice that at each print interval, several variables describing the performance of the TWT are printed to the output file. Which variables are printed is determined by the variable, NPR, in the input data file. The four options available are described and illustrated in this section.



```
                 TWT1D  10:58 AM  MON., 20  JAN., 1986
                   RESULTS FOR INPUT FILE SAMPLE.IN

N  FREQ.(gHz)   INPUT PWR.(dBm)   REL. PHASE(rad)
1    18.0000        10.0000           0.0000

                 CATHODE CURRENT(Amps)=    .2800
                    HELIX VOLTAGE(Volts)=  10200.00
               EFFECTIVE VOLTAGE(Volts)=   9808.02
               NORMALIZED BEAM VELOCITY=    .1959
                      HELIX PITCH(tpi)= 19.180
               PLASMA FREQUENCY(rads/s)=    .1141E+11
   PLASMA FREQUENCY REDUCTION FACTOR=       .5987
                  SPACE CHARGE PARAMETER=   .7993
                  MEAN HELIX RADIUS(in)=    .03790
                       BEAM FILL FACTOR=    .5000
          CIRCUIT PRINT INTERVAL(in)=       .2000
           NUMBER OF ELECTRON DISKS= 16
               NUMBER OF ATTENUATORS=  1
                   NUMBER OF CHANGES=  3

N  NORM. PHASE VEL.(Vp/c)   AXIAL IMP.(ohms)   GAIN PARAMETER
1          .1864                 3.6000             .0318

    Z        PDB       B      THETA      DPDB      DTDZ      ETA       D
   .200     9.874     1.60    -.505     -.628    -1.0610    .0000    .0527
   .400     9.909     1.60   -1.035      .176    -1.1227    .0000    .05
   .600    10.32      1      -1.587     2.088    -1.1447    .0000    .05

RESTART INFORMATION FOR Y=    7.009

     A           DA          THETA        DTHETA
 .7872E-01    .3605E-01   -.1192E+02   -.1975E+01

ELECTRON PHASE POSITIONS
 .1200E+02
 .1241E+02
 .1280E+02
 .1318E+02
 .1353E+02
 .1388E+02
 .1422E+02
 .1458E+02
 .1494E+02
 .1532E+02
 .1572E+02
 .1614E+02
 .1656E+02
 .1700E+02
 .1743E+02
 .1786E+02

ELECTRON VELOCITIES
  .7256E-01    .4444E-01    .6205E-02   -.3519E-01
 -.7122E-01   -.9445E-01   -.1012E+00   -.9222E-01
 -.7145E-01   -.4343E-01   -.1196E-01    .1990E-01
  .4919E-01    .7247E-01    .8605E-01    .8667E-01

    4.680    32.075     1.60   -12.394     9.327    -1.1459    .0006    .0527
    4.880    33.921      .45   -12.335     9.232     -.4336    .0009    .05

    9.350    54.127     2.41   -28.754     1.430    -1.9667    .0906    .0548
    9.500    54.201     2.41   -29.300      .497    -1.2292    .0921    .0548

                        ENDING PRINT OUT
                  SATURATED POWER (dBm)=   54.201
               GAIN AT SATURATION (dB)=    44.201
           SATURATION EFFICIENCY (%)=       9.213
            LENGTH AT SATURATION (in)=      9.500
               SMALL SIGNAL GAIN (dB)=    44.707

                 TWT1D   1:35 PM  MON., 20  JAN., 1986
```

Fig. A.6. TWT1D output file.

NPR = 1

This is the printout option which will most often be used by tube designers. It prints out the following variables.

Z Axial position (in)

PDB Power (dBm)

B Pierce's velocity parameter

THETA The phase angle of the circuit wave (radians)

DPDB The partial derivative of power with respect to position

DTDZ The partial derivative of phase angle with respect to position

ETA The efficiency

D Pierce's loss parameter

NPR = 2

This option prints out all the variables in NPR = 1, as well as information about energy in the beam and in the circuit wave. This can help indicate whether or not the computer simulation is a good representation of the physical process. The variables are:

EBEAM Power lost by the beam, normalized by the total beam power

EWAVE Power contained in the circuit wave and in the field between the circuit and the beam

EBLNC $(|EBEAM| - |EWAVE|)/(|EBEAM| + |EWAVE|)$
If the simulation is reasonable, EBLNC will be near zero. A "good" value for EBLNC is on the order of 0.1. However, this parameter does not account for the circuit wave power lost in attenuators so, in those regions, EBLNC will not be near zero.

NPR = 3

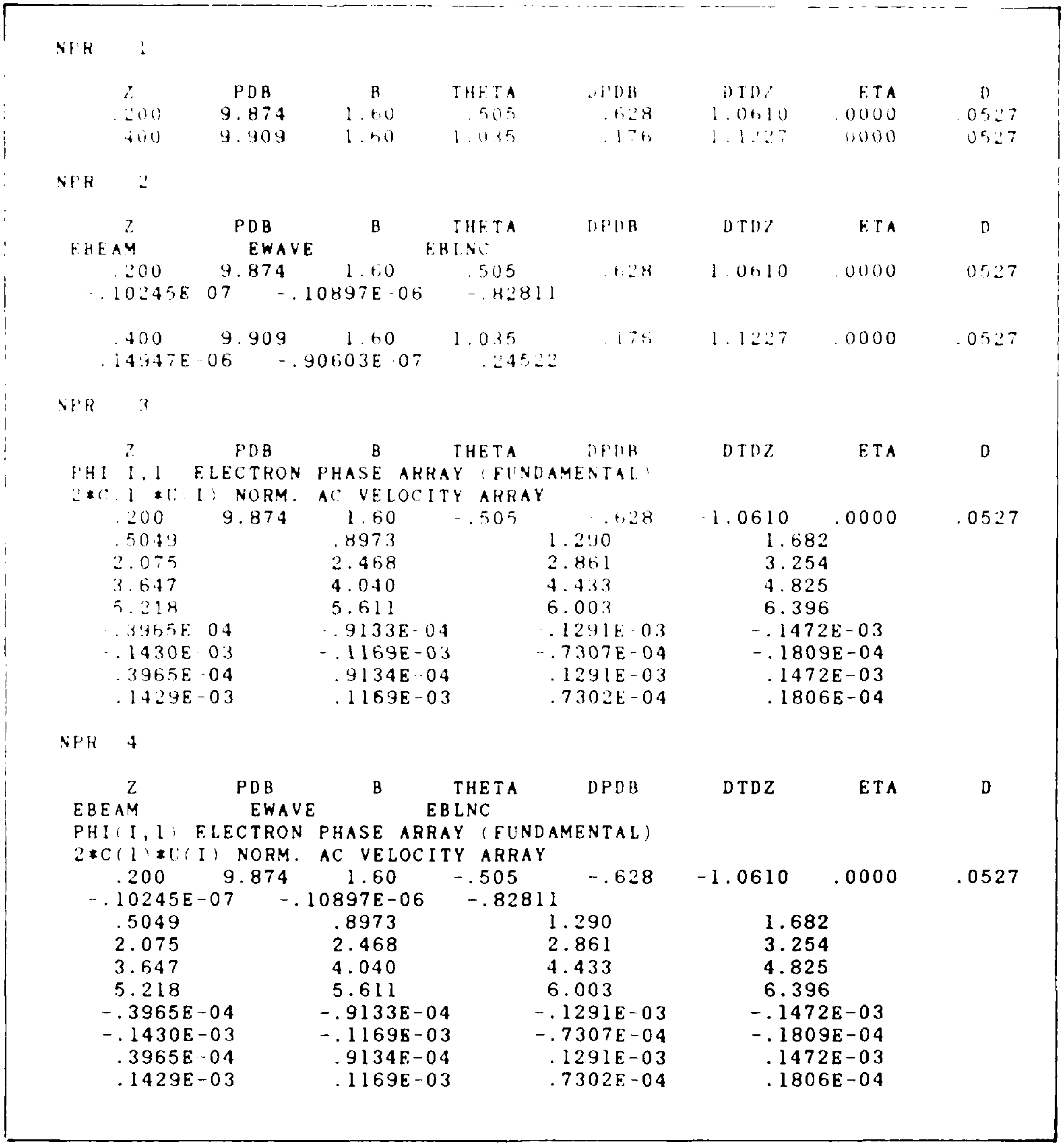
This option includes all the variables in NPR = 1, as well as information for creating electron flight diagrams and Cutler charts. The variables are:

PHI(i,1)   Electron phases for the fundamental frequency

2*C(1)*U(i)   Normalized electron velocities

NPR = 4

This option will print out all the variables listed above. It generates a lot of data! All of the NPR options are illustrated in Fig. A.7.



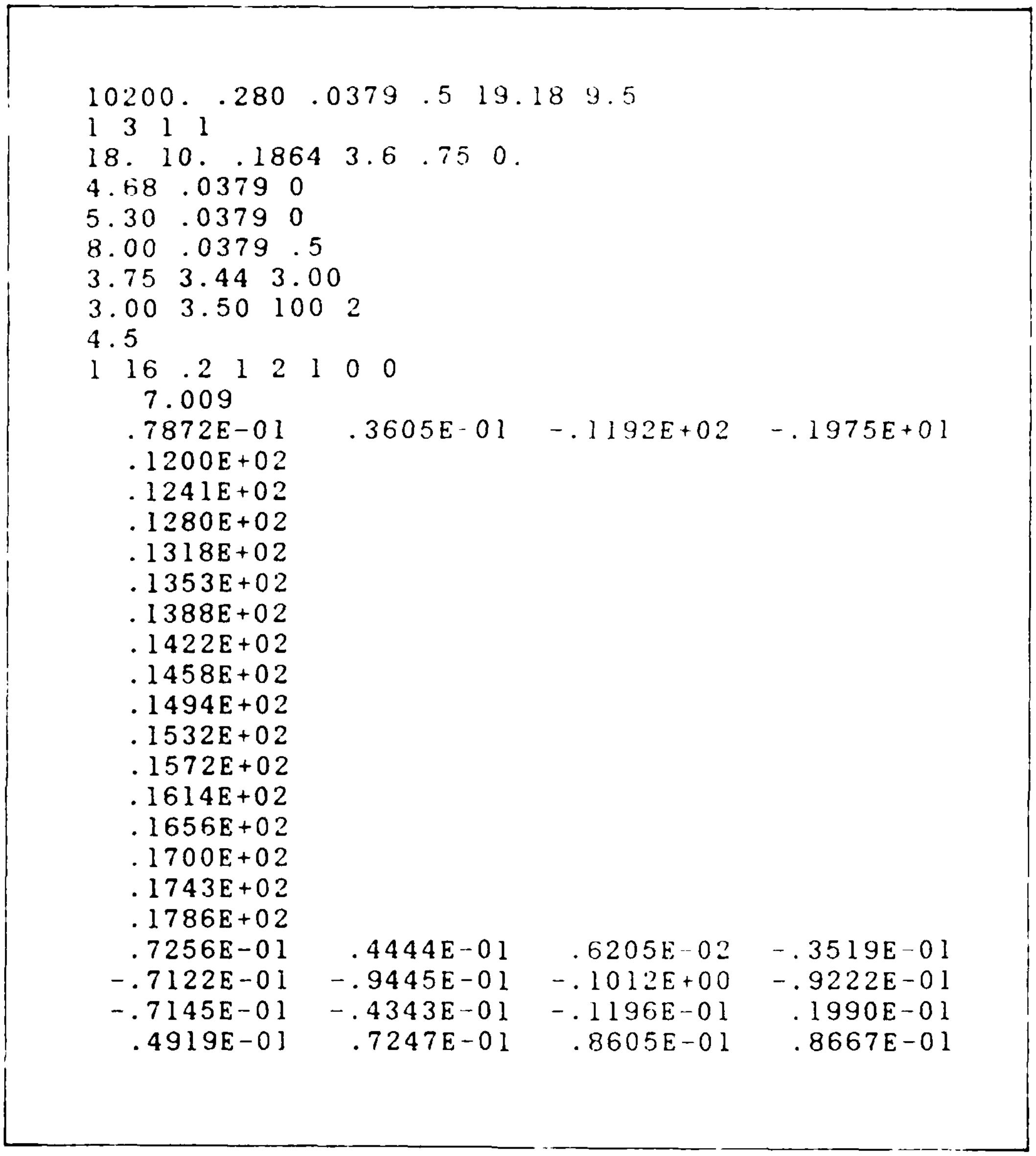
```
NPR    1

       Z          PDB         B       THETA        DPDB        DTDZ        ETA         D
     .200       9.874       1.60       .505        .628      1.0610      .0000      .0527
     .400       9.909       1.60      1.035        .176      1.1227      .0000      .0527

NPR    2

       Z          PDB         B       THETA        DPDB        DTDZ        ETA         D
 EBEAM            EWAVE           EBLNC
     .200       9.874       1.60       .505        .628      1.0610      .0000      .0527
   -.10245E-07    -.10897E-06     -.82811

     .400       9.909       1.60      1.035        .178      1.1227      .0000      .0527
    .14947E-06    -.90603E-07      .24522

NPR    3

       Z          PDB         B       THETA        DPDB        DTDZ        ETA         D
 PHI(I,1) ELECTRON PHASE ARRAY (FUNDAMENTAL)
 2*C(1)*U(I) NORM. AC VELOCITY ARRAY
     .200       9.874       1.60      -.505       -.628     -1.0610      .0000      .0527
     .5049              .8973               1.290               1.682
    2.075              2.468               2.861               3.254
    3.647              4.040               4.433               4.825
    5.218              5.611               6.003               6.396
   -.3965E-04         -.9133E-04          -.1291E-03          -.1472E-03
   -.1430E-03         -.1169E-03          -.7307E-04          -.1809E-04
    .3965E-04          .9134E-04           .1291E-03           .1472E-03
    .1429E-03          .1169E-03           .7302E-04           .1806E-04

NPR  4

       Z          PDB         B       THETA        DPDB        DTDZ        ETA         D
 EBEAM            EWAVE           EBLNC
 PHI(I,1) ELECTRON PHASE ARRAY (FUNDAMENTAL)
 2*C(1)*U(I) NORM. AC VELOCITY ARRAY
     .200       9.874       1.60      -.505       -.628     -1.0610      .0000      .0527
   -.10245E-07    -.10897E-06     -.82811
     .5049              .8973               1.290               1.682
    2.075              2.468               2.861               3.254
    3.647              4.040               4.433               4.825
    5.218              5.611               6.003               6.396
   -.3965E-04         -.9133E-04          -.1291E-03          -.1472E-03
   -.1430E-03         -.1169E-03          -.7307E-04          -.1809E-04
    .3965E-04          .9134E-04           .1291E-03           .1472E-03
    .1429E-03          .1169E-03           .7302E-04           .1806E-04
```

Fig. A.7. TWT1D printout options.

[illegible]. Restarting a Run

Often an engineer will want to examine the effect of different output helices on a tube's performance. TWT1D allows the user to do this without having to rerun the input helix in each case.

If a run is to be restarted, the input file for the initial run should have the number of restarts, NUMR set to 1, and the position of the restart printout, ZRST, set to the position of the restart. The output file for that run will contain all the information needed to restart the run. A restart printout at 4.5 inches is illustrated in Fig. A.6. To restart the run, the data from the restart must be appended to the original input file, and the restart data input switch, RSTART, set to 1. Figure A.8 shows the input file for a restarted run. Note that all the character data have been removed from the restart data.



```
10200. .280 .0379 .5 19.18 9.5
1 3 1 1
18. 10. .1864 3.6 .75 0.
4.68 .0379 0
5.30 .0379 0
8.00 .0379 .5
3.75 3.44 3.00
3.00 3.50 100 2
4.5
1 16 .2 1 2 1 0 0
   7.009
  .7872E-01    .3605E-01   -.1192E+02   -.1975E+01
  .1200E+02
  .1241E+02
  .1280E+02
  .1318E+02
  .1353E+02
  .1388E+02
  .1422E+02
  .1458E+02
  .1494E+02
  .1532E+02
  .1572E+02
  .1614E+02
  .1656E+02
  .1700E+02
  .1743E+02
  .1786E+02
  .7256E-01    .4444E-01    .6205E-02   -.3519E-01
 -.7122E-01   -.9445E-01   -.1012E+00   -.9222E-01
 -.7145E-01   -.4343E-01   -.1196E-01    .1990E-01
  .4919E-01    .7247E-01    .8605E-01    .8667E-01
```

Fig. A.8. Input file for restarting a run.

MISSION
of
Rome Air Development Center

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control, Communications and Intelligence (C^{3}I) activities. Technical and engineering support within areas of competence is provided to ESD Program Offices (POs) and other ESD elements to perform effective acquisition of C^{3}I systems. The areas of technical competence include communications, command and control, battle management, information processing, surveillance sensors, intelligence data collection and handling, solid state sciences, electromagnetics, and propagation, and electronic, maintainability, and compatibility.